高等院校信息管理与信息系统专业系列教材

数据仓库与数据挖掘教程

陈文伟 编著



清华大学出版社

高等院校信息管理与信息系统专业系列教材

数据仓库与数据挖掘教程

陈文伟 编著

清华大学出版社
北 京

内 容 简 介

数据仓库与数据挖掘都是从数据资源提取信息和知识进行辅助决策。由于数据资源丰富，数据仓库与数据挖掘辅助决策效果十分显著。

本书系统介绍数据仓库原理、联机分析处理、数据仓库设计与开发、数据仓库的决策支持应用，数据挖掘原理、信息论的决策树方法、集合论的粗糙集方法、关联规则、公式发现、神经网络、遗传算法、文本挖掘与 Web 挖掘，以及数据仓库与数据挖掘的发展。

本书对数据仓库的系统介绍，在于突出决策支持的本质。对数据挖掘的各类方法均介绍了它们的理论基础和实现方法，并通过例子进行了说明。

本书的特点是从数据仓库和数据挖掘的兴起与演变来说明它们的本质，通过实例来解释它们的原理，这样便于读者学习和掌握，适于本科生和研究生使用。

图书在版编目（CIP）数据

数据仓库与数据挖掘教程/陈文伟编著．—北京：清华大学出版社，2006.8
（高等院校信息管理与信息系统专业系列教材）
ISBN 978-7-302-13154-0

Ⅰ．数…　Ⅱ．陈…　Ⅲ．①数据库系统－高等学校－教材 ②数据采集－高等学校－教材
Ⅳ．①TP311.13 ②TP274

中国版本图书馆 CIP 数据核字(2006)第 059604 号

责任编辑：范素珍
责任印制：王秀菊

出版发行：清华大学出版社　　**地　　址：**北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编：**100084
社　总　机：010-62770175　　**邮　　购：**010-62786544
投稿与读者服务：010-62795954，jsjjc@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京鑫海金澳胶印有限公司
装 订 者：
经　　销：全国新华书店
开　　本：185×260　　**印　　张：**18.5　　**字　　数：**443 千字
印　　次：2009 年 7 月第 5 次印刷
印　　数：10501～13500
定　　价：25.00 元

产品编号：019389-02TP

出版说明

20世纪三四十年代，一直摸索着前进的计算技术与刚走向成熟的电子技术结缘。这一结合，不仅孕育了新一代计算工具——电子计算机，还产生了当时谁也没有料到的巨大效应：电子计算机——这种当初为计算而开发出来的工具，很快就超出计算的范畴，成为“信息处理机”的代名词；人类开始能够高效率地开发并利用信息；信息对人类社会的作用得以有效地发挥，并逐步超过材料和能源，成为人类社会的重要支柱；信息产业急剧增长，信息经济高度发展，社会生产力达到了新的高度；人们的信息化意识不断加强，人类在信息资源方面更加激烈地竞争，社会发展走上信息化轨道。

文化是时代的精髓，是特定的人群在一定的历史时期、一定的地域范围对其生产和生活模式、思维和行为方式的觉悟和理性化，它伴随着人类创造和使用工具能力的提高而不断发展。文者，经天纬地也；化者，变化、改变、造化、习俗、风气也。也可以说，文化作为社会的人们在生产和生活中思维和行为方式的理性化，是文治和教化的结果。因此，文化具有区域性、群体性和时代性。在信息时代的帷幕刚刚拉开、新时代的气息开始弥漫社会各个角落的20世纪70年代，先觉们就已开始创办以加速信息化的进程为宗旨、以培养信息资源开发人才为目标的信息管理与信息系统专业。

从与信息有关的学科纵向来看，信息管理与信息系统专业处于信息学、信息技术、信息管理、信息经济、信息社会学这个层次结构的中间，它下以信息学和信息技术为基础，上与信息经济和信息社会学相联系。从其涉及的学科横向来看，它处在管理学、信息科学与技术和有关专业领域的交叉点上。它对技术有极高的要求，又要求对组织有深刻的理解，对行为有合理的组织，反映了科学与人文融合的特点。这种交叉与融合正是信息管理与信息系统专业最重要的特征，是其他的学科或专业难以取代和涵盖的。

我国的信息管理与信息系统专业创建于20世纪70年代末。在近20年的时间里，已发展到151个点，成为培养信息化人才的重要领域。其发展速度之快、影响之深远已令世人和学术界刮目相看。然而作为一个新的、特别是与各行各业关系极为密切的专业，其课程体系、教学内容以及教学方法、手段，都要经历一个逐步完善、逐步成熟的过程，其教材体系的建设更需要较长期的实践和探索。没有这样一个过程，具有专业特点、符合中国实际的教材体系是不会建立的。近20年来，大家一直在课程体系的完善和建设有自己专业特点的教材方面不断进行探讨。1991年，全国10所财经类院校的经济信息管理专业的负责人在太原召开第一次研讨会。以后，1993年在大连、1995年在武汉、1997年在烟台，又有更多的院校参加到了这一研讨之中。这些研讨活动得到了国家教委有关部门的赞许和支持。通过研讨，大家在建设具有专业特点的教材体系、改变简单照搬其他专业教材上取得了共识。在武汉会议之后，即着手进行系列教材的编写工作。经协商，由张基温教授担任主编，由魏晴宇教授、陈禹教授担任顾问。

这套教材是我国信息管理与信息系统专业的第一套教材。尽管编写者为它付出了巨大

的辛劳，但在实践中我们也深深地感到了时代的鞭策和工作的难度。一方面，席卷全球的信息化大潮已经使信息、信息管理、信息系统成为全社会关注的热点，人们对其期望和要求越来越高；另一方面，在世纪之交的今天，作为现代社会先导技术的信息技术和相关学科的更新速度在不断加快，多种社会因素相互渗透、相互影响，新情况、新问题给专业的建设带来很多的困难。当然，这些对我们专业的发展和建设也是一种动力和机遇。为此，在这套教材问世之际，我们再一次表示一个心愿：希望与全国的同行共勉，在教材和专业建设上齐心协力，做出更大贡献。也由于如上种种原因，这套教材不会是完整的，也不会是完美的，一定存在这样那样的不足或错误，我们将会不断补充，不断修改，不断完善。任何建设性意见都是我们非常期盼的。为此，这一套教材将具有充分的开放性：每一本教材都是一个原型，每一条建设性意见都将会被采纳，并享有自己的知识产权。

全国高等院校计算机基础教育研究会
财经信息管理专业委员会
信息管理与信息系统专业系列教材编委会

前　言

数据仓库(data warehouse,DW)是利用数据资源提供决策支持。它比利用模型资源辅助决策更有效,而且辅助决策的范围更宽。由于在现实中,数据大量存在,而且在迅速地增长,只要将面向应用(事务驱动)的数据库重新组织转变为面向决策分析的数据仓库,就可以帮助决策者从不同的视角,通过综合数据分析掌握现状;通过多维数据分析发现各种存在的问题;通过对数据层次的钻取找出问题产生的原因;通过历史数据预测未来。由于数据仓库辅助决策效果明显,数据仓库已经从 20 世纪 90 年代中期兴起,经过几年的发展,迅速形成了潮流。

数据挖掘(data mining,DM)是从数据中挖掘出信息和知识,是从人工智能的机器学习(machine learning,ML)中发展起来的。机器学习是让计算机模拟人的学习方法获取知识。机器学习中的大量学习方法已经引入到数据挖掘中。数据挖掘也是 20 世纪 90 年代中期兴起的。正是由于数据挖掘具有获取知识的能力,目前各数据仓库均将数据挖掘作为数据仓库的前端分析工具,用于提高数据仓库的决策支持能力。

数据仓库、数据挖掘和联机分析处理(on line analytical processing,OLAP)结合起来的新决策支持系统是以数据驱动的决策支持系统。而传统决策支持系统(decision support system,DSS)是以模型和知识驱动的决策支持系统,是由模型库系统、知识库系统、数据库系统和人机交互系统组成的。新决策支持系统利用的是数据资源,而传统决策支持系统利用的是模型资源和知识资源,它们两者辅助决策的方式和效果均不相同。新决策支持系统并不能代替传统决策支持系统,它们是相互补充的。新决策支持系统与传统决策支持系统结合起来形成的综合决策支持系统将是决策支持系统发展的新方向。

数据仓库、数据挖掘、联机分析处理等结合起来也称为商业智能(business intelligence,BI)。商业智能是一种新的智能技术,区别于人工智能(artificial intelligence,AI)和计算智能(computational intelligence,CI)。人工智能采用的技术是符号推理,符号推理过程形成了概念的推理链。计算智能采用的技术是计算推理,模拟人和生物的模糊推理、神经网络计算和遗传进化过程。商业智能是从数据仓库和数据挖掘中获取信息和知识,对变化的商业环境提供决策支持。商业智能是目前企业界正在大力推广的知识管理(knowledge management,KM)的基础。

作者于 1997 年 6 月 30 日在《计算机世界》报上发表了一组关于数据开采(数据挖掘)的文章,最早向国内学者介绍了数据挖掘概念和技术。作者又于 1998 年 6 月 15 日在《计算机世界》报上发表了一组关于数据仓库与决策支持系统的文章,在介绍基于数据仓库的决策支持系统上,提出了将基于数据仓库的决策支持系统和传统决策支持系统结合的综合决策支持系统,在国内产生了一定的影响。

本书的特点是从数据仓库和数据挖掘的兴起与演变来说明它们的本质,通过例子来解释它们的原理,既系统地介绍了数据仓库和数据挖掘的概念和技术,又介绍了它们之间的关

系，以及今后的发展。

在数据仓库的章节中，重点介绍数据仓库原理、联机分析处理、数据仓库设计与开发、数据仓库的决策支持应用。在数据挖掘的章节中重点介绍信息论方法、集合论方法、公式发现、神经网络和遗传算法，这些数据挖掘方法在现实中应用较广泛。由于数据挖掘的基础理论涉及面较宽，建议在本科生教学中对信息论原理和集合论方法只讲定义和例子，对神经网络和遗传算法只讲公式和应用，省略原理的深层内容和公式的推导。这些省略的内容适合研究生教学。

由于作者从事数据仓库与数据挖掘工作多年，并得到过国家自然科学基金项目的资助。在书中还介绍了作者领导的课题组完成的IBLE决策规则树方法、FDD公式发现系统、遗传分类学习系统GCLS等。本书也包含了作者提出的综合决策支持系统概念和可拓数据挖掘概念及理论，这些内容适合研究生学习和参考。

欢迎和广大读者进行交流，共同为促进我国数据仓库和数据挖掘的发展而努力。

参加本书录入的有毕季明、廖建文、赵健、徐怡峰、田昊等同志，在此表示感谢！

陈文伟

目　录

第1章　数据仓库与数据挖掘概述

1.1　数据仓库的兴起

1.1.1　从数据库到数据仓库

由数据库(DB)发展到数据仓库(DW)主要在于如下几点。

- 数据太多,信息贫乏(data rich, information poor):随着数据库技术的发展,企事业单位建立了大量的数据库,数据越来越多,而辅助决策信息却很贫乏,如何将大量的数据转化为辅助决策信息成了研究的热点。
- 异构环境数据的转换和共享:由于各类数据库产品的增加,异构环境的数据也随之增加,如何实现这些异构环境数据的转换和共享也成了研究的热点。
- 利用数据进行事务处理转变为利用数据支持决策:数据库用于事务处理,若要达到辅助决策,则需要更多的数据。例如,如何利用历史数据的分析来进行预测。对大量数据的综合得到宏观信息等均需要大量的数据。

数据仓库概念提出后,在不到几年的时间内就得到了迅速的发展。数据仓库产品也不断出现并陆续进入市场。

1. 数据库用于事务处理

数据库存储大量的共享数据,作为数据资源用于管理业务中的事务处理,已经成为了成熟的信息基础设施。

数据库中存放的数据基本上是保存当前数据,随着业务的变化随时更新数据库中的数据。例如,学生数据库,随着新生的入校,数据库中要增加新学员的数据记录;随着毕业学生的离校,数据库中要删除这些学员的数据记录。数据库总是保持当前的数据记录。

不同的管理业务需要建立不同的数据库。例如,银行中储蓄业务要建立储蓄数据库,记录所有储蓄用户的存款及使用信息;信用卡业务要建立信用卡数据库,记录所有用户信用卡的存款及使用信息;贷款业务要建立贷款数据库,记录贷款用户的贷款及使用信息。

数据库是为事务处理需求设计和建立的,从而使计算机在事务处理上发挥极大的效果。但是,数据库在帮助人们进行决策分析时就显得不适应了。例如,银行想了解用户的经济状态(收入与支出情况)以及信誉如何(是否超支,还贷情况等)?是否继续贷款给他?单靠一个数据库是无法完成这种决策分析的。必须将储蓄数据库、信用卡数据库、贷款数据库集中起来,对某一个人进行全面分析,才能准确了解他的存款及收支情况、信用卡使用情况以及贷款和还贷情况。这样,银行才能有效地决定是否给此人继续贷款。

同时使用3个数据库进行操作并非是一件简单的事,由于3个管理业务各自独立,在建

立数据库时对同一个人可能使用了不同的编码，对于他的姓名可能有的用汉字，有的用汉语拼音，有的用英文。这为使用 3 个数据库地共同进行决策分析带来了困难。

2. 数据仓库用于决策分析

随着决策分析的需求扩大，兴起了支持决策的数据仓库。它是以决策主题需求集成多个数据库，重新组织数据结构，统一规范编码，使其有效地完成各种决策分析。

从数据库到数据仓库的演变，体现了以下几点。

(1) 数据库用于事务处理，数据仓库用于决策分析

事务处理功能单一，数据库完成事务处理的增加、删除、修改、查询等操作。决策分析要求数据较多。数据仓库需要存储更多的数据，不需要修改数据，主要提取综合数据的信息，以及分析预测数据的信息。

(2) 数据库保持事务处理的当前状态，数据仓库既保存过去的数据又保存当前的数据

数据库中数据随业务的变化一直在更新，总保存当前的数据，如学生数据库。数据仓库中数据不随时间变化而变化，但保留大量不同时间的数据，即保留历史数据和当前数据。

(3) 数据仓库的数据是大量数据库的集成

数据仓库的数据不是数据库的简单集成，而是按决策主题，将大量数据库中数据进行重新组织，统一编码进行集成。如银行数据仓库数据是由储蓄数据库、信用卡数据库、贷款数据库等多个数据库按“用户”主题进行重新组织、编码和集成而建立的。可见，数据仓库的数据量比数据库的数据量大得多。

(4) 对数据库的操作比较明确，操作数据量少。对数据仓库操作不明确，操作数据量大

一般对数据库的操作都是事先知道的事务处理工作，每次操作(增加、删除、修改、查询)涉及的数据量也小，如一个或几个记录数据。

对数据仓库的操作都是根据当时决策需求临时决定而进行的。如比较两个地区某个商品销售的情况。该操作所涉及的数据量很大，不是几个记录数据，而是两个地区多个商店的某商品的所有销售记录。

3. 数据库与数据仓库对比

数据库与数据仓库的对比如表 1.1 所示。

表 1.1　数据库与数据仓库对比

数据库	数据仓库
面向应用	面向主题
数据是详细的	数据是综合的或提炼的
保持当前数据	保存过去和现在的数据
数据是可更新的	数据不更新
对数据操作是重复的	对数据的操作是启发式的
操作需求是事先可知	操作需求是临时决定的

续表

数据库	数据仓库
一个操作存取一个记录	一个操作存取一个集合
数据非冗余	数据时常冗余
操作比较频繁	操作相对不频繁
查询的是原始数据	查询的是经过加工的数据
事务处理需要的是当前数据	决策分析需要过去、现在的数据
很少有复杂的计算	很多复杂的计算
支持事务处理	支持决策分析

1.1.2 从 OLTP 到 OLAP

1. 联机事物处理(on line transaction processing,OLTP)

联机事物处理是在网络环境下的事务处理工作,利用计算机网络技术,以快速的事务响应和频繁的数据修改为特征,使用户利用数据库能够快速地处理具体的业务。OLTP 是事务处理从单机到网络环境发展的新阶段。OLTP 应用要求多个查询并行,以便将每个查询分布到一个处理器上。

OLTP 的特点在于事务处理量大,但事务处理内容比较简单且重复率高。大量的数据操作主要涉及的是一些增加、删除、修改、查询等操作。每次操作的数据量不大且多为当前的数据,OLTP 的数据组织的数据模型采用实体-关系(E-R)模型。

OLTP 处理的数据是高度结构化的,涉及的事务比较简单,数据访问路径是已知的,至少是固定的。事务处理应用程序可以直接使用具体的数据结构,如表、索引等。

OLTP 面对的是事务处理操作人员和低层管理人员。

在过去三十多年中,OLTP 系统发展的目标就是能够处理大量的数据。每时间单位能够处理更多的事务,能支持更多的并发用户,且有更好的系统健壮性。大型的系统每秒能够处理 1000 个以上的事务。有些系统,像机票预订系统,每秒能够处理的事务峰值可以达到 2 万个。

数据库存储的数据量很大,经常每天要处理成千上万的事务,OLTP 在查找业务数据时是非常有效的。但是为高层领导者提供决策分析时,则显得力不从心。

2. 联机分析处理(on line analytical processing,OLAP)

关系数据库之父 E. F. Codd 在 1993 年认为,联机事务处理已经不能满足终端用户对数据库决策分析的需要,决策分析需要对多个关系数据库共同进行大量的综合计算才能得到结果。为此,他提出了多维数据库和多维分析的概念,即联机分析处理概念。关系数据库是二维数据(平面),多维数据库是空间立体数据。

近年来,人们利用信息技术生产和搜集数据的能力大幅度提高,大量的数据库被用于商业管理、政府办公、科学研究和工程开发等,这一势头仍将持续发展下去。于是,一个新的挑战被提出来:在信息爆炸的时代,信息过量几乎成为人人需要面对的问题。如何才能不被

信息的汪洋大海所淹没,从中及时发现有用的知识或者规律,提高信息利用率呢?要想使数据真正成为一个决策资源,只有充分利用它为一个组织的业务决策和战略发展服务才行,否则大量的数据可能成为包袱,甚至成为垃圾。OLAP是解决这类问题的最有力的工具之一。

OLAP专门用于支持复杂的分析操作,侧重对分析人员和高层管理人员的决策支持,可以应分析人员的要求快速、灵活地进行大数据量的复杂处理,并且以一种直观易懂的形式将查询结果提供给决策制定人,以便他们准确掌握企业(公司)的经营情况,了解市场需求,制定正确方案,以增加效益。OLAP软件以它先进的分析功能和以多维形式提供数据的能力,正作为一种支持企业关键商业决策的解决方案而迅速崛起。

OLAP的基本思想是决策者从多方面和多角度以多维的形式来观察企业的状态和了解企业的变化。

3. OLTP与OLAP的对比

OLAP是以数据仓库为基础,其最终数据来源与OLTP一样均来自底层的数据库系统,但由于二者面对的用户不同,OLTP面对的是操作人员和低层管理人员,OLAP面对的是决策人员和高层管理人员,因而数据的特点与处理也明显不同。

OLTP和OLAP是两类不同的应用,它们的各自特点见表1.2所示。

表1.2　OLTP与OLAP对比表

OLTP	OLAP
数据库数据	数据库或数据仓库数据
细节性数据	综合性数据
当前数据	历史数据
经常更新	不更新,但周期性刷新
一次性处理的数据量小	一次性处理的数据量大
对响应时间要求高	响应时间合理
用户数量大	用户数量相对较少
面向操作人员,支持日常操作	面向决策人员,支持决策需要
面向应用,事务驱动	面向分析,分析驱动

1.1.3　数据字典与元数据

1. 数据库的数据字典

数据字典是数据库中各类数据描述的集合,在数据库设计中占有很重要的地位。数据字典通常包括数据项、数据结构、数据流、数据存储和处理过程5个部分,其中数据项是数据的最小组成单位。若干个数据项可以组成一个数据结构。数据字典通过对数据项和数据结构的定义来描述数据流、数据存储的逻辑内容。

(1) 数据项

数据项是不可再分的数据单位。对数据项的描述通常包括数据项名、数据项含义说明、数据类型、长度、取值范围和取值含义等。

(2) 数据结构

数据结构反映了数据之间的组合关系。一个数据结构可以由若干个数据项组成,也可以由若干个数据结构组成。数据结构的描述通常包括数据结构名、含义说明和数据项等。

(3) 数据流

数据流是数据结构在系统内传输的路径,对数据流的描述通常包括数据流名、说明、数据流来源、数据流去向和平均流量等。其中“数据流来源”是说明该数据流来自哪个过程。“数据流去向”是说明该数据流将到哪个过程去。“平均流量”是指单位时间(如每天)传输的次数。

(4) 数据存储

数据存储是数据结构保存数据的地方,数据存储的描述通常包括数据存储名、说明、编号、输入的数据流、输出的数据流、数据量、存取频度和存取方式。其中“存取频度”指每小时或每天或每周存取几次、每次存取多少数据等信息。“存取方式”包括是批处理还是联机处理;是检索还是更新;是顺序检索还是随机检索等。另外,“输入的数据流”要指出其来源,“输出的数据流”要指出其去向。

(5) 处理过程

处理过程一般用判定表或判定树来描述。数据字典中只需要描述处理过程的说明性信息,通常包括处理过程名、说明、输入、输出和处理。其中“处理”中主要说明该处理过程的功能及处理要求。

可见,数据字典是关于数据库中数据的描述,而不是数据本身。

2. 数据仓库的元数据

数据仓库远比数据库复杂。在数据仓库中引入了“元数据”的概念,它不仅是数据仓库的字典,而且还是数据仓库本身信息的数据。

元数据(meta data)定义为关于数据的数据(data about data),即元数据描述了数据仓库的数据和环境。

元数据在数据仓库中不仅定义了数据仓库有什么,还指明了数据仓库中信息的内容和位置,刻画了数据的抽取和转换规则,存储了与数据仓库主题有关的各种商业信息,而且整个数据仓库的运行都是基于元数据的,如数据的修改、跟踪、抽取、装入、综合以及使用等。由于元数据遍及数据仓库的所有方面,已成为整个数据仓库的核心。

数据仓库的元数据除对数据仓库中数据的描述(数据仓库字典)外,还有以下 3 类元数据。

(1) 关于数据源的元数据

数据仓库的数据源包含了很多不同的数据结构,为数据仓库选取的数据元素字段长度和数据类型而有所不同。为数据仓库挑选数据时,得将记录拆分,并将来自不同源文件的记录的某些部分组合起来,还要解决编码和字段长度不同的问题。当将这些信息传递给最终用户的时候,必须把这些数据与原始数据联系起来。

(2) 关于抽取和转换的元数据

这类元数据包含了源数据系统的数据抽取方法、数据抽取规则,以及抽取频率等数据转

换的所有信息。

(3) 关于最终用户的元数据

最终用户元数据是数据仓库的导航图,使最终用户可以从数据仓库中找到自己需要的信息。

1.1.4 数据仓库的定义与特点

数据仓库的概念是由 W. H. Inmon 在《建立数据仓库(Building the Data Warehouse)》一书中提出的。数据仓库的提出是以关系数据库、并行处理和分布式技术为基础的信息新技术。

从目前的形势看,数据仓库技术已紧跟 Internet,成为信息社会中获得企业竞争优势的又一关键技术。

1. 数据仓库定义

(1) W. H. Inmon 对数据仓库的定义

数据仓库是面向主题的、集成的、稳定的、不同时间的数据集合,用于支持经营管理中决策制定过程。

(2) SAS 软件研究所的观点

数据仓库是一种管理技术,旨在通过通畅、合理、全面的信息管理,达到有效的决策支持。

从数据仓库的定义可以看出,数据仓库是为决策支持服务的,而数据库是为事务处理服务的。

2. 数据仓库特点

从数据仓库的定义可以看出数据仓库的特点如下。

(1) 数据仓库是面向主题的

主题是数据归类的标准,每一个主题基本对应一个宏观的分析领域。例如,保险公司的数据仓库的主题为客户、政策、保险金和索赔等。

基于应用的数据库组织则完全不同,它的数据只是为处理具体应用而组织在一起的。保险公司按应用组织的数据库是汽车保险、生命保险、健康保险和伤亡保险等。

(2) 数据仓库是集成的

数据进入数据仓库之前,必须经过加工与集成。对不同的数据来源进行统一数据结构和编码。统一原始数据中的所有矛盾之处,如字段的同名异义、异名同义、单位不统一和字长不一致等。总之,将原始数据结构做一个从面向应用到面向主题的大转变。

(3) 数据仓库是稳定的

数据仓库中包括了大量的历史数据。数据经集成进入数据仓库后是极少或根本不更新的。

(4) 数据仓库是随时间变化的

数据仓库内的数据时限在 5～10 年,故数据的键码包含时间项,并标明数据的历史时

期,以便适合决策分析时进行时间趋势分析。

而数据库只包含当前数据,即存储某一时间的正确的有效数据。

(5) 数据仓库中的数据量很大

通常的数据仓库的数据量为 10GB 级,相当于一般数据库 100MB 的 100 倍,大型数据仓库是一个 TB(1000GB)级数据量。

数据仓库中数据量的比重是:索引和综合数据占 2/3,原始数据占 1/3。

(6) 数据仓库软硬件要求较高

① 需要一个巨大的硬件平台;

② 需要一个并行的数据库系统。

1.2 数据挖掘的兴起

1.2.1 从机器学习到数据挖掘

学习是人类具有的智能行为,主要在于获取知识。机器学习是研究使计算机模拟或实现人类的学习行为,即让计算机通过算法自动获取知识。机器学习是人工智能领域中的重要研究方向。

20 世纪 60 年代开始了机器学习的研究。比较典型的成果有:Rosenblate 的感知机,它是最早用神经网络进行模式识别的方法;Sammel 的西洋跳棋程序,它用线性表达式的启发式方法,通过多次人机对弈,自动修改表达式中的系数,使程序逐渐聪明,该程序竟然达到胜过开发者和州冠军的成绩。

20 世纪 80 年代,机器学习取得了较大的成果。Michelski 等人的 AQ11 系统(1980)能从大量病例中归纳出大豆病症的判断规则。AQ11 是一个很成功的归纳学习方法;Quiulan 的 ID3(1983)决策树方法,影响很大,实用效果很强;Langley 等人的 BACON 系统(1987)能重新发现物理学的大量规律;Rumelhart 等人研制的反向传播神经网络 BP 模型(1985)为神经网络的学习开创了一个新阶段。

这些显著成果的出现,使"机器学习"逐渐形成了人工智能的主要学科方向之一。1980 年在美国召开了第一届国际机器学习学会研讨会,1984 年《机器学习》杂志问世。

中国在 1987 年召开了第一届全国机器学习研讨会。1989 年成立了中国人工智能学会机器学习学会。中国学者洪家荣研制的 AE1 系统(1985)采用了扩张矩阵方法;钟鸣等人研制的 IBLE 方法(1992)利用信道容量建立决策规则树,识别效果比 ID3 方法更高。本书作者研制的 FDD 经验公式发现系统(1998),能发现含初等函数或复合函数的经验公式,发现的公式比 BACON 系统发现的公式范围更宽。

1989 年美国召开了第一届知识发现(knowledge discovery in database,KDD)国际学术会议,从数据库中发现知识形成了新概念。KDD 研究的问题有:定性知识和定量知识的发现;知识发现方法;知识发现的应用等。

1995 年在加拿大召开了第一届知识发现和数据挖掘(data mining,DM)国际学术会议。由于把数据库中的"数据"形象地比喻成矿床,"数据挖掘"一词很快流传开来。

数据挖掘是知识发现中的核心工作，主要研究发现知识的各种方法和技术。而这些方法和技术主要来自于机器学习。由于数据挖掘的发展，出现了一些新的数据挖掘方法，如大型数据库中关联规则的挖掘，以及利用粗糙集进行属性约简和规则获取等。

数据挖掘兴起时主要是在数据库中挖掘知识，随着数据仓库的出现和发展，很快将数据挖掘技术和方法用于数据仓库。典型的啤酒与尿布的故事（该两商品同时出售的出现概率很高）就是在数据仓库中挖掘出的关联知识。

1.2.2 数据挖掘的含义

按《人工智能辞典》的定义：信息是数据中所蕴涵的意义。知识是人们对客观世界的规律性认识。

数据库中每个数据记录的内涵代表了该记录的信息。而数据挖掘是从数据库中所有数据记录中归纳总结出知识。知识的数量大大少于数据记录量。这些知识代表了数据库中数据信息的规律，即用少量的知识能够覆盖数据库中所有的记录。

例如，人口数据库中存储各国人口的记录，它将是一个庞大的数据库。但是，通过数据挖掘，可以得出形式化表示的规则知识：

$$(头发=黑色) \vee (眼睛=黑色) \rightarrow 亚洲人$$

其中∨表示“或”；→表示“蕴涵”，规则知识表示为：“若（条件）则（结论）”，即表示：若头发是黑色或者眼睛是黑色的人，则他是亚洲人。

该知识代表了亚洲人的特点，即覆盖了所有亚洲人的记录。

知识的获得是通过数据挖掘算法，如 AQ11 方法和 ID3 方法等经过计算得到的。

1.2.3 数据挖掘与 OLAP 的比较

1. OLAP 的多维分析

OLAP 是在多维数据结构上进行数据分析的。对多维数据进行分析是复杂的。一般从多维数据取出（切片、切块）二维或三维数据进行分析，或对层次的维进行钻取操作，向下钻取获得更详细的数据，向上钻取获得更综合的数据。

OLAP 要适应大量用户同时使用同一批数据，适应于不同地理位置的分散化的决策。OLAP 的功能和算法包括聚合、分配、比率、乘积等描述性的建模功能。

OLAP 平时需要查询大量的日常商业活动信息，如每周的布匹购买量、每周布匹的内部库存以及布匹的销售量等。OLAP 更需要查询商业活动的变化情况，如每周布匹购买量的变化值、衣服生产量的变化值、衣服销售价格的变化等。这些变化值对经理制定决策更重要。

经理往往从查询出的变化值中，通过 OLAP 追踪查询，找出存在的原因。例如，经理看到利润小于预计值的时候，可能会深入到各个国家查看整个产品利润情况。这样，他可能发现有些国家的利润明显低于其他国家，于是他自然就会查看这些国家中不同产品组的利润情况，总的目标就是寻找一些比较异常的数据来解释某个现象。经过一番观察之后，就会发现非直接成本在这些国家明显偏高。进一步对这些非直接成本分析，可以发现近期对于某

些产品的赋税明显增加，从而明显影响了最终的利润。这种分析查询要求时间响应快。

以上是 OLAP 的典型应用，通过商业活动变化的查询发现的问题，经过追踪查询找出问题出现的原因，达到辅助决策的作用。

2. 数据挖掘

OLAP 是在带层次的维度和跨维度进行多维数据分析的。数据挖掘则不同，是以变量和记录为基础进行分析的。

数据挖掘任务在于聚类（如神经网络聚类）、分类（如决策树分类）、预测等。这些是带有探索性的建模功能。

数据挖掘在于寻找不平常的且有用的商业运作模型。考察数据的不同类型或者找出变量之间的关系。数据挖掘需要查看海量数据，主要是详细数据和历史数据。为此经常将数据仓库中的数据复制到一个专门的存储器上，对数据的挖掘分析可能要花去大量的时间，即不要求快速分析。数据挖掘人员有时并不能精确地知道什么是必须分析的，有时数据挖掘一无所获。但是，有时通过数据挖掘会发现意外的、无价的信息"金块"。例如，如果能够确定一个高价值的客户或可能离开的客户特征，就可以要求公司采取措施保留这些客户，这比从竞争对手那儿重新争取曾经失去的客户费用少得多。

1.2.4 数据挖掘与统计学

1. 统计学的发展过程

统计学是一门有悠久历史的学科。统计学开始于 17、18 世纪，与国家政治有紧密的关系。英国 W. Petty（1623—1682 年）的《政治算术》一书中第一次用计量和比较的方法，对英国与法、意、荷等国进行国力比较。J. Graunt（1620—1674 年）通过统计计算，发现男女人数占人口数的比例大致相等、出生儿中男婴比例稍高、婴幼儿的死亡率较大等规律性的现象。

17 世纪，B. Pascal 等人提出"概率"概念，用来描述某一事件发生的可能性。18 世纪，在观测天体运动时会有误差产生，虽然多次测量，由于有误差，得到的总是和真值不同的值。高斯（Gauss，1777—1855 年）提出误差值落在(a,b)区间的概率等于该区间上正态分布曲线下的面积，称误差服从正态分布（高斯分布）。比利时的凯特勒（A. Quetelet，1796—1874 年）称"支配着社会现象的法则和方法是概率论"。

近代统计学重视社会调查。通过对全部对象（总体）进行调查，为制定计划和决策提供依据，如果对总体的某些分布情况有一定把握，就不必搞全面调查，可以搞部分调查，即抽样调查，由部分推断全部。概率论和数理统计理论起着重要的作用。现在各国在经济统计、国事调查、社会调查、收视率调查、民意测验等采用的几乎都是抽样调查。

现代统计学从线性到非线性、从低维到高维、从显在到潜在、从连续到离散等方面有较完备的理论和方法。统计软件包 SPSS、SAS 等已经普及，统计工作基本上利用计算机来完成。

2. 统计学中应用于数据挖掘的内容

(1) 常用统计

在大量数据中求最大值、最小值、总和、平均值等。

(2) 相关分析

通过求变量间的相关系数来确定变量间的相关程度。

(3) 回归分析

建立回归方程(线性或非线性)以表示变量间的数量关系,再利用回归方程进行预测。

(4) 假设检验

在总体存在某些不确定情况时,为了推断总体的某些性质,提出关于总体的某些假设,对此假设利用置信区间来检验,即任何落在置信区间之外的假设判断为“拒绝”,任何落在置信区间之内的假设判断为“接受”。

(5) 聚类分析

将样品或变量进行聚类的方法,具体方法是把样品中每一个样品看成是 m 维空间的一个点,聚类是把“距离”较近的一个点归为同一类,而将“距离”较远的点归为不同的类。

(6) 判别分析

建立一个或多个判别函数,并确定一个判别标准。对未知对象利用判别函数将它划归某一个类别。

(7) 主成分分析

主成分分析是把多个变量化为少数的几个综合变量,而这几个综合变量可以反映原来多个变量的大部分信息。

主成分分析的一种推广是因子分析,即用少数几个因子(F_i)去描述许多变量(X_j)之间的关系。变量(X_j)是可以观测的显在变量,而因子(F_i)是不可观测的潜在变量。

3. 统计学与数据挖掘的比较

统计学主要是对数量数据(数值)或连续值数据(如年龄、工资等)进行数值计算(如初等运算)的定量分析,得到数量信息。如常用统计量(最大值、最小值、平均值、总和等)、相关系数、回归方程等。

数据挖掘主要对离散数据(如职称、病症等)进行定性分析(覆盖、归纳等),得到规则知识。例如,如果某人的眼睛是黑的或者头发是黑的,则可以认为他是亚洲人。

在统计学中有聚类分析和判别分析,它们与数据挖掘中的聚类和分类相似。但是,采用的标准不一样,统计学的聚类采用的“距离”是欧式距离,即两点间的坐标(数值)距离。而数据挖掘的聚类采用的“距离”是海明距离,即属性取值是否相同,相同者距离为 0,不相同者距离为 1。

总之,统计学与数据挖掘是有区别的,但是,它们之间是相互补充的。不少数据挖掘的著作中均把统计学的不少方法引入到数据挖掘中,与将机器学习中不少方法引入到数据挖掘中一样,作为从数据获取知识的一大类方法。

虽然统计学的不少方法可以归入到数据挖掘中,但统计学仍然是一门独立的学科。

1.3 数据仓库和数据挖掘的结合

1.3.1 数据仓库和数据挖掘的区别与联系

1. 数据仓库与数据挖掘的区别

数据仓库是在数据库的基础上发展起来的。它将大量的数据库的数据按决策需求进行重新组织，以数据仓库的形式进行存储，将为用户提供辅助决策的随机查询、综合信息以及随时间变化的趋势分析信息等。

数据仓库是一种存储技术，其数据存储量是一般数据库的100倍，包含大量的历史数据、当前的详细数据以及综合数据。它能适应不同用户对不同决策需要提供所需的数据和信息。

数据挖掘是从人工智能机器学习中发展起来的。它研究各种方法和技术，从大量的数据中挖掘出有用的信息和知识。最常用的数据挖掘方法是统计分析方法、神经网络方法和机器学习中研究的方法。数据挖掘中采用机器学习的方法有归纳学习方法（如覆盖正例排斥反例方法，如AQ系列算法、决策树方法等）、遗传算法、发现学习算法（如公式发现系统BACON）等。

利用数据挖掘的方法和技术从数据仓库中挖掘的信息和知识，反映了数据仓库中数据的规律性。用户利用这些信息和知识来指导和帮助决策。例如，利用分类规则来预测未知实体的类别。

2. 数据仓库与数据挖掘的关系

数据仓库与数据挖掘都是决策支持新技术，但它们有着完全不同的辅助决策方式。数据仓库中存储着大量辅助决策的数据，为不同的用户随时提供各种辅助决策的随机查询、综合信息或趋势分析信息。数据挖掘是利用一系列算法挖掘数据中隐含的信息和知识，让用户在进行决策中使用。

数据仓库和数据挖掘可以结合起来。在数据仓库系统的前端分析工具中，数据挖掘是其中重要工具之一。它可以帮助决策用户挖掘数据仓库的数据中隐含的规律性。

数据挖掘用于数据仓库实现决策支持，具体表现为：

（1）预测客户购买倾向；

（2）客户利润贡献度分析；

（3）分析欺诈行为；

（4）销售渠道优化分析等。

数据仓库和数据挖掘的结合对支持决策会起更大的作用。

3. 数据仓库中数据存储特点

数据挖掘兴起时是针对数据库，随着数据仓库的兴起和发展，由于数据仓库不同于数据

库,数据挖掘也随之发生变化。

(1) 数据存储方式的不同

数据库的数据存储是按照管理业务中事物处理项目的要求而存放的。

数据仓库的数据存储是按决策分析需求而存放的。这种需求是以决策主题为对象的,典型的主题是客户。这样,在数据仓库中客户数据需要从多个数据库集成而来,如银行数据仓库需要从储蓄、信用卡、贷款等不同数据库中,对同一客户的数据抽取并集成在一起,以便完成对该客户的分析。

(2) 数据存储的数据量的不同

数据库的数据存储量相对数据仓库的数据存储量小得多。从上面的例子可以看出,以客户主题建立数据仓库的数据量是储蓄、信用卡、贷款3个数据库的数据量的总和。按一般的统计,数据仓库的数据量是数据库数据量的100倍。数据仓库的数据量比数据库的数据量大这么多在于:

① 数据仓库中的数据(近期基本数据)是数据库中数据按决策主题重新组织并集成而来;

② 数据仓库中数据还需要保留大量的历史数据,用于预测分析;

③ 数据仓库为了给不同级别管理者提供各种决策分析的数据,需要对近期基本数据进行轻度综合和高度综合,这些综合数据在数据仓库中占据了不小的比重。

近期基本数据、历史数据、综合数据三者的数据相加,使数据仓库的数据量远远大于数据库中的数据量。

(3) 数据存储的结构不同

由于数据仓库的数据量远大于数据库的存储量,数据库的关系型二维(平面)存储格式不能适应数据仓库。数据仓库的数据存储结构采用多维的超立方体结构形式。数据仓库的数据存储结构采用星型模型或者多维立体数据库形式。

4. 数据仓库中数据挖掘特点

数据仓库的最大应用在于扩展市场,制定营销策略,争取更多的客户。

(1) 数据挖掘从数据仓库中挖掘的信息

数据挖掘应用于数据仓库后,能挖掘更深层次上的信息。如:

- 哪些商品一起销售好?(利用关联分析)
- 偏爱某类商品的客户特征是什么?(利用聚类分析)
- 还有哪些客户具有上述特征?(利用类比分析)
- 哪些商业事务处理可能有欺诈性?(利用神经网络)
- 高价值客户的共同点是什么?(利用分类分析)

典型的例子是通过数据挖掘对高价值客户以及可能离开的客户进行挖掘,得出它们的特征,这样就让公司作出决策,达到保留这些高价值的客户和争取可能离开的客户,从而提高公司的利润。

(2) 数据仓库为数据挖掘提出了新要求

① 数据挖掘需要可扩展性

数据挖掘对数据仓库的应用一般使用的数据是详细数据，不用综合数据，因为综合数据“平滑”了数据间的差别，从而无法发现单个数据项目之间的微妙相关性。

数据仓库中的数据随着时间的推移，数据逐渐增长。这样，数据挖掘方法就应该具有可扩展性，能够处理递增的数据量。

② 数据挖掘方法需要能挖掘多维知识

数据仓库中的数据模型是多维数据组织，它不同于数据库的二维数据组织。数据挖掘应用到数据仓库时需要能挖掘多维数据知识。

例如，对数据库的关联分析只能得到同一个商品维中不同商品之间的关联关系。到数据仓库中的关联分析就应该能对多维数据寻找它们的关联关系，即除不同商品的关联外，还要找出商品与商店或时间等不同维之间的关联关系。

1.3.2 基于数据仓库的决策支持系统

在建立数据仓库之前，利用数据库完成决策分析时，由于决策者不能明确表明到底需要什么具体数据来帮助辅助决策，一开始会提出一个粗糙的需求，由 IT(信息技术)人员编写专门程序从数据库中抽取数据，形成所需的报告。决策者根据这个报告会马上想起需要更多的数据，需要提供新的报告。IT 人员重新编写程序抽取新的数据，完成新的报告。

由于决策的不明确性，对数据抽取的多样性，包括不同时间的抽取以及不同角度的抽取，形成的分析报告会造成不同的结果，甚至于矛盾的结果。例如，一个 IT 人员提出的分析报告说企业的业绩下降了 15%，另一个 IT 人员提出的分析报告说企业的业绩上升了 10%。这两个结论不但不吻合，而且相去甚远。这让决策者很难相信报告结论的正确性，也无法帮助决策。

从而认识到在数据库的基础上编写专门的程序，获取信息辅助决策是不成功的。人们把这种方式建立的决策支持系统认为是失败的。

为了建立随时提取销售量最好的产品名单；告诉出现问题的地区；并能分析出现问题的原因；对比各种数据；显示最大的利润等辅助决策信息的决策支持系统，数据仓库成了惟一可行的解决方案。

数据仓库对整个企业各部门的数据进行统一和综合，这实际上是决策支持的一次革新。企业可以用它来取得各个重要方面的数据与分析结果。例如商品利润、市场分析和风险管理等，从而改善企业的自身管理。举例来说，数据仓库用户可以立即得到其单位当前所处地位的准确报告，了解其公司面临的风险，包括各项事务及整个企业所有业务面临的风险，并对市场和法规条例需要迅速作出反应。

数据仓库的决策支持功能有：

(1) 对当前和历史数据完成查询和报表处理；

(2) 可以用不同方法进行“如果，将怎样(what-if)”分析；

(3) 可以查询细节，查询综合，并能深入追踪查询处理；

(4) 认清过去的发展趋势，并将其应用于对未来结果的分析。

数据仓库是为辅助决策而建立的，单依靠数据仓库达到辅助决策的能力是有限的。数据仓库中有大量的综合数据，这些数据为决策者提供了综合信息，即反映企业或部门的宏观

状况。数据仓库保存大量历史数据，这些数据通过预测模型计算可以得到预测信息。

综合信息与预测信息是数据仓库所获得的辅助决策信息。

数据仓库中增加联机分析处理和数据挖掘等分析工具，能较大地提高辅助决策能力。联机分析处理对数据仓库中的数据进行多维数据分析，即多维数据的切片、切块、旋转、钻取等，只有通过分析更详细的数据，才能得到更深层中的信息和知识。如节假日销售的影响，某日的促销活动的影响等，这些信息在综合数据中是反映不出来的。数据挖掘技术能获取关联知识、时序知识、聚类知识、分类知识等。数据挖掘技术对数据仓库中的数据进行挖掘，才能获取更多的辅助决策信息和知识。

数据仓库和联机分析处理及数据挖掘结合的决策支持系统，是以数据仓库为基础的，称为基于数据仓库的决策支持系统，其结构图如图 1.1 所示。

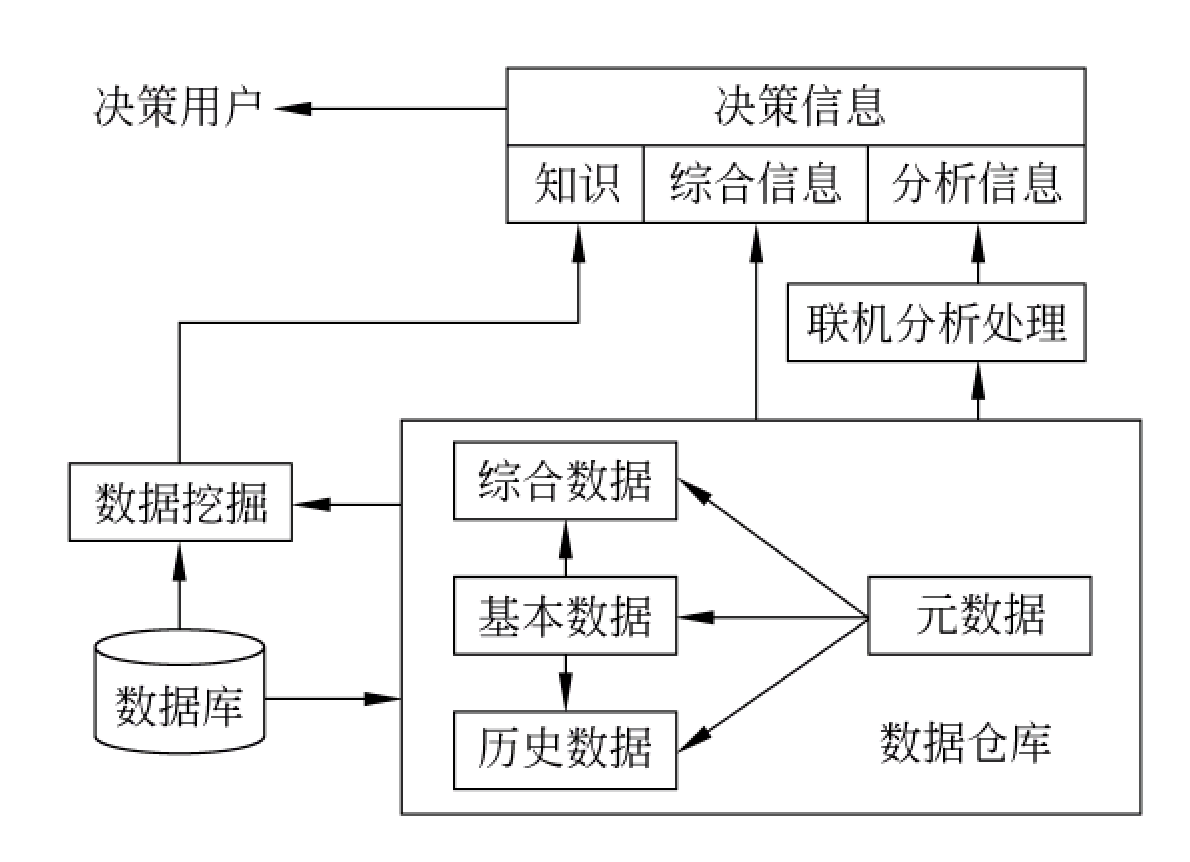


图 1.1　基于数据仓库的决策支持系统结构

概括地说：基于数据仓库的决策支持系统是从数据仓库的数据中获取辅助决策的信息和知识，为决策提供支持。

基于数据仓库的决策支持系统区别于 20 世纪 80 年代出现的基于模型的决策支持系统和 20 世纪 90 年代兴起的智能决策支持系统。把基于模型和知识的智能决策支持系统称为传统的决策支持系统，而把基于数据仓库的决策支持系统称为新决策支持系统。

1.3.3　数据仓库与商业智能

1. 商业智能的概念

商业智能是 20 世纪 90 年代中期提出的。商业智能以数据仓库为基础，通过联机分析处理和数据挖掘技术帮助企业领导者针对市场变化的环境，作出快速、准确的决策。

商业智能与新决策支持系统从组成和目标来看是一致的。但是，商业智能是一种技术，新决策支持系统是解决实际决策问题的一个系统。可以理解为：新决策支持系统是利用商业智能技术来解决实际决策问题的系统。

数据仓库、联机分析处理与数据挖掘组成的商业智能所体现的智能行为在于，能够解决市场环境中随机变化的决策问题。由于市场千变万化，每次需要解决的决策问题都不相同。这种解决随机出现的问题需要利用智能的手段。商业智能所提供的智能手段表现为联机分

析处理的任意切片、切块和钻取，以及利用数据挖掘技术所获得的知识。

2. 商业智能辅助制定更好更快的决策

公司需要制定的决策有两类：由高层管理者制定宏观的战略决策；基层人员在日常事务中制定决策。战略决策有：投资哪个项目？哪些业务需要分离还是合并？制定销售策略等。事务决策有：销售员决定是否给一个客户折扣；生产经理决定是否投产一个新产品以满足客户需求；市场营销专家决定是否要进行新一轮的直接邮购活动；采购经理决定是否买更多的材料等。这些事务决策只具有“战术”意义，不会影响到业务运作的基础，从总体效果看，其重要性并不亚于企业高级管理人员作出的重大决策，但会直接影响企业的成败。这些决策很少是通过决策分析而作出的，大多靠的是经验、积累的知识和惯常的做法。提高企业日常工作中的决策质量，将直接对企业的成本和营业收入产生影响。

商业智能改进企业决策过程，表现如下几个方面：

（1）信息共享

有了商业智能系统就可以实现信息共享，用户可以迅速找到所需要的数据，通过对数据进行钻取分析以达到目标。例如，某公司通过商业智能系统跟踪商品的质量管理，能及时发现问题，而不是一个星期后查阅各种报告来发现问题。时间的节省以及产品质量的提高，不仅降低了企业的成本，也给公司带来了更多的收入。

（2）实时反馈分析

商业智能的运用能够使员工随时看到工作进展程度，并且了解一个特定的行为对现实目标的效用。如果员工们都能看到自己的行为如何提升或者影响了业绩，那么也就不需要过于复杂的激励体系了。

例如，朋斯卡物流公司，司机的激励机制与其驾驶表现，如每英里的耗油量和损耗程度等成本控制方面的因素相关联。通过电子商业智能系统，公司的主控计算机就能根据司机出车行驶的里程计算出每加仑汽油能支持的里程数，然后再把数据传输到数据仓库，员工们通过数据仓库就可以分析提高绩效的可能性，即发现汽车保养或司机驾驶习惯如何调整来达到业绩目标，提高业务水平并创造更多的价值。

（3）鼓励用户找出问题的根本原因

根据初步得到的答案而采取的行动可能未必正确，因为初步的探究往往没有发现根本问题的所在。要找出根本原因就需要对与成功或失败相关的诸多因素进行深度分析。

通过企业商业智能系统，能够找到某部门业绩糟糕或者出色的根本原因，只要不断地追问“为什么？为什么？”这个过程可能是从分析一个报告开始，比如每季度的销售情况，每个答案引出一个新问题，采取钻取或分析方法，就能把最根本的原因找出来。例如，通过企业商业智能系统，制衣商发现他们推出的市场促销活动效果不理想。在分析诸多数据后，制衣商开始把价格跟市场需求进行灵活挂钩。结果，该制衣商减少了存货时间，提高了存货管理的效率，营运资本、销售、利润等几项主要业绩指标也明显好转。

（4）使用主动智能

在数据仓库中设定预警机制，一旦出现超过预警条件的数据，就自动通过各种设备，比如电子邮件、传呼、手机等通知用户。这种主动智能使用户及时决断，并采取相应措施。

(5) 实时智能

企业采用真正的实时智能,将大大提高运营效率、降低成本、提高服务质量。例如朋斯卡物流公司认识到需要一个商业智能系统来实时监控和智能管理运输及物流业务。该系统掌握了很多信息,把货物运载量维持在一个最高的水平,帮助客户更快地把货物从 A 地送到 B 地。企业商业智能系统能实时跟踪卡车的货物装载量。如果一辆卡车的装载量只有一半,公司根据商业智能系统发出指令让该车调整路线,再装载一些货物。这样该系统使公司的所有营业收入上升了很多。

习　　题

1. 数据库与数据仓库的本质差别是什么?
2. 从数据库发展到数据仓库的原因是什么?
3. 举例说明数据库与数据仓库的不同。
4. 说明 OLTP 概念和 OLAP 概念。
5. OLTP 如何在网络数据库上进行事务处理?
6. 说明 OLTP 与 OLAP 的主要区别。
7. 数据库中数据字典包括哪些内容?
8. 元数据的定义是什么?
9. 元数据与数据字典的关系是什么?
10. 数据仓库的定义是什么?
11. 数据仓库的特点有哪些?
12. 说明机器学习如何形成人工智能的学科方向。
13. 说明数据挖掘的含义。
14. OLAP 多维分析如何辅助决策? 举例说明。
15. 数据挖掘与 OLAP 有什么不同?
16. 通过例子说明统计学的价值。
17. 说明统计学应用于数据挖掘中所包含的内容。
18. 说明统计学与数据挖掘的不同。
19. 说明数据仓库与数据挖掘的区别与联系。
20. 数据挖掘应用于数据库与数据挖掘应用于数据仓库有什么不同?
21. 举例说明数据挖掘从数据仓库中挖掘的信息有哪些?
22. 数据仓库为数据挖掘提出了哪些新要求?
23. 数据仓库与联机分析处理、数据挖掘在决策支持方面有什么不同?
24. 基于数据仓库的决策支持系统的组成是什么?
25. 画出基于数据仓库的决策支持系统结构图。
26. 说明基于数据仓库的决策支持系统与传统决策支持系统有什么区别。
27. 商业智能概念是什么?
28. 如何理解商业智能与基于数据仓库的决策支持系统的区别和联系?
29. 商业智能在哪些方面改进企业决策过程?

第2章　数据仓库原理

2.1　数据仓库结构体系

2.1.1　数据仓库结构

数据仓库是在原有关系型数据库基础上发展形成的，但不同于数据库系统的组织结构形式，它从原有的业务数据库中获得的数据形成当前基本数据层，经过综合后形成轻度综合数据层，轻度综合数据再经过综合后形成高度综合数据层。W. H. Inmon 在《建立数据仓库》一书中给出数据仓库的结构如图 2.1 所示。数据仓库结构包括当前基本数据(current detail data)、历史基本数据(older detail data)、轻度综合数据(lightly summarized data)、高度综合数据(highly summarized data)和元数据。

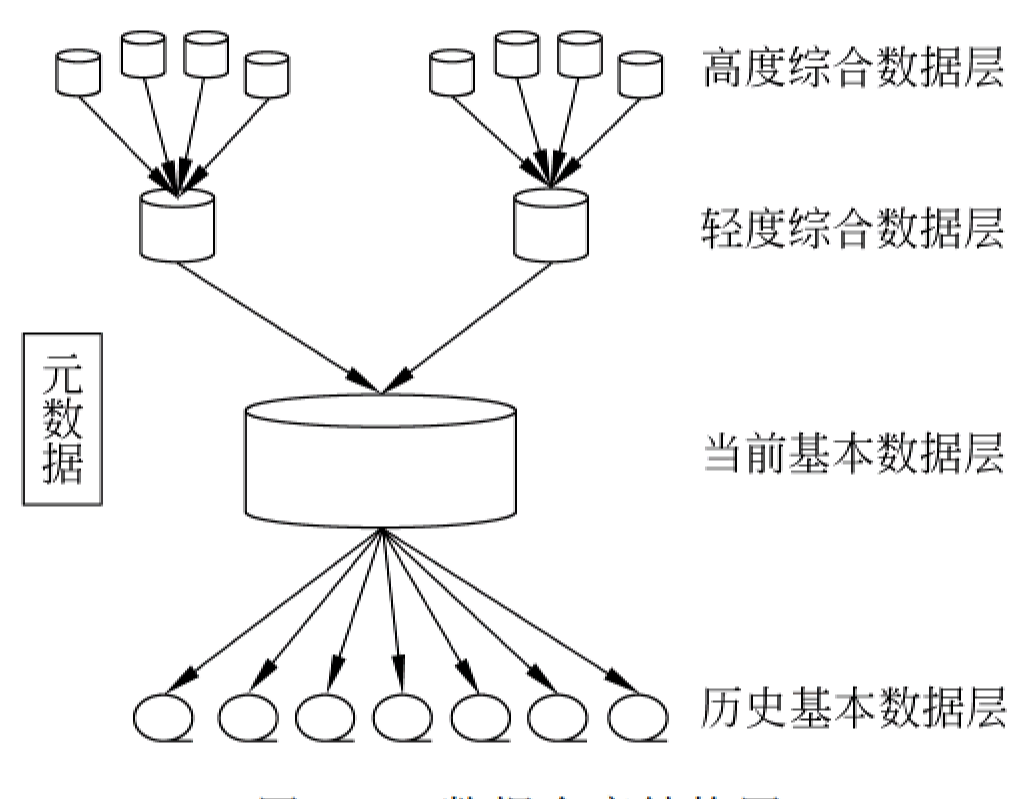


图 2.1　数据仓库结构图

当前基本数据是最近时期的业务数据，是数据仓库用户最感兴趣的部分，数据量大。当前基本数据随时间的推移，由数据仓库的时间控制机制转为历史基本数据，一般被转存于介质中，如磁带等。轻度综合数据是从当前基本数据中提取出来的，设计这层数据结构时会遇到“综合处理数据的时间段选取，综合数据包含哪些数据属性(attributes)和内容(contents)”等问题。最高一层是高度综合数据层，这一层的数据十分精练，是一种准决策数据。

整个数据仓库的组织结构是由元数据来组织的，不包含任何业务数据库中的实际数据信息。元数据在数据仓库中扮演了重要的角色，包括如下信息：数据仓库的目录信息；数据从业务环境向数据仓库环境转换时的目录内容；指导从当前基本数据到轻度综合数据，以及轻度综合数据到高度综合数据的综合算法的选择。

在数据库中只存储当前的详细数据。而数据仓库除存储按主题组织起来的当前详细数据外，还需要存储综合数据，这是为适应决策需求而增加的。在数据库中需要得到综合数据时，采用临时编制程序对详细数据进行综合。在数据仓库中并不采取临时计算的方式得到

综合数据，而是在用户提出需要综合数据之前，就预先将可能需要的综合数据计算好，存入综合数据层中，这种综合数据层在用户查询时，能迅速提供给用户。为此，在建数据仓库时，要分析好各类用户可能需要的哪些综合数据，并将这些综合数据都存在综合数据层中。

综合数据与详细数据是不同“粒度”的数据。粒度是指数据仓库的数据单元中保存数据的细化或综合程度的级别。细化程度越详细，粒度级就越低。

不同粒度的数据的存储数据量差距很大。例如，在低粒度级（详细数据）上，每次电话都详细记录下来，一个月每位顾客平均有 200 条记录，总共需要 40 000 个字节。在高粒度级（综合数据），每位顾客只有一个记录，大约只需要 200 个字节。

高粒度级不仅只需要少得多的字节存放数据，而且只需要较少的索引项。这样的数据存储效率较高。

在数据仓库环境中，粒度之所以是设计数据仓库的一个重要方面，不只因为它影响了存放在数据仓库中的数据量的大小，它同时也影响数据仓库所能回答的查询类型。当提高数据粒度时（综合数据），数据所能回答查询的能力将会随之降低。而很小粒度的数据（详细数据）可以回答任何问题，但在高粒度的数据上（综合数据），可以回答的问题就很少。

例如，提出如下查询：“张三上星期是否给他在外地的女友打了电话？”在低粒度级上这个问题是可以回答的，这需要查阅大量的记录，该查询最终总是可以确定的。然而在高粒度级上就无法明确回答这个问题，因为在高粒度级上只存放有张三打出电话总数，不能确定其中是否有一个电话是打往外地女友的。

但是，在进行决策分析时，很少对单个事件进行查询，通常是针对某个数据集合进行处理的（这在数据仓库环境中是常见的）。例如，提出综合查询：“上个月人们从广州打出的长途电话平均多少个？”在决策分析中，这种类型的查询非常多。该查询既可以在高粒度上也可以在低粒度上进行处理。但在回答这个问题时，在不同粒度级上所使用的资源具有相当大的差别。在低粒度级上回答这个问题需要查询每一个记录，使用大量的资源来回答这个问题。在高粒度级上，包括了足够的细节（如包括每个顾客打出长途电话的次数），使用高粒度级数据的效率就会高很多。例如，在轻度综合级上电话记录如下，将能使用较小的资源回答以上问题：

三月份，李四，电话数量：46 个，电话平均长度：10 分钟，长途电话数：12 个。

在数据仓库中存储多种粒度数据（详细层、轻度综合层、高度综合层等）是为提高决策分析的效果。大部分决策分析处理是针对存储效率高的轻度综合层数据进行的。当需要分析更低的细节级数据（占 5%或者更少的可能）时，可以到详细数据层数据上进行。在详细数据层上访问数据是昂贵的、复杂的。

2.1.2 数据集市及其结构

数据仓库是企业级的，能为整个企业各个部门的运行提供决策支持手段；而数据集市则是部门级的，一般只能为某个局部范围内的管理人员服务，因此也称为部门级数据仓库（departmental data warehouse）。

1. 数据集市(data marts)的产生

数据仓库工作范围和成本常常是巨大的。信息技术部门必须对所有的用户并以全企业的眼光对待任何一次决策分析。这样,就形成了代价很高的、时间较长的大项目。

人们认识到了提供更紧密集成的、拥有完整图形接口并且价格吸引人的工具——数据集市,就应运产生。

目前,全世界对数据仓库总投资的一半以上均集中在数据集市上。

2. 数据集市概念

数据集市是一种更小、更集中的数据仓库,为公司提供分析商业数据的一条廉价途径。

数据集市是指具有特定应用的数据仓库,主要针对某个具有战略意义的应用或者具体部门级的应用,支持用户利用已有的数据获得重要的竞争优势或者找到进入新市场的具体解决方案。

数据集市有两种,即独立的数据集市(independent data mart)和从属的数据集市(dependent data mart)。

3. 数据集市与数据仓库的差别

(1) 数据仓库是基于整个企业的数据模型建立的,它面向企业范围内的主题。而数据集市是按照某一特定部门的数据模型建立的,由于每个部门有自己特定的需求,因此对数据集市的期望也不一样。

(2) 部门的主题与企业的主题之间可能存在关联,也可能不存在关联。数据仓库中存储整个企业内非常详细的数据,而数据集市中的数据的详细程度要低一些,包含概要和累加数据要多一些。

(3) 数据集市的数据组织一般采用星型模型。大型数据仓库的数据组织,如 NCR 公司采用第三范式。

4. 数据集市的特性

(1) 规模是小的;
(2) 特定的应用;
(3) 面向部门;
(4) 由业务部门定义,设计和开发;
(5) 由业务部门管理和维护;
(6) 快速实现;
(7) 价格较低廉;
(8) 投资快速回收;
(9) 工具集的紧密集成;
(10) 更详细的、预先存在的数据仓库的摘要子集;
(11) 可升级到完整的数据仓库。

5. 两种数据集市结构

(1) 从属数据集市

从属数据集市的逻辑结构如图 2.2 所示。

所谓从属,是指它的数据直接来自于中央数据仓库。显然,这种结构仍能保持数据的一致性。一般为那些访问数据仓库十分频繁的关键业务部门建立从属的数据集市,这样可以很好地提高查询的反应速度。

(2) 独立数据集市

独立数据集市的逻辑结构如图 2.3 所示。

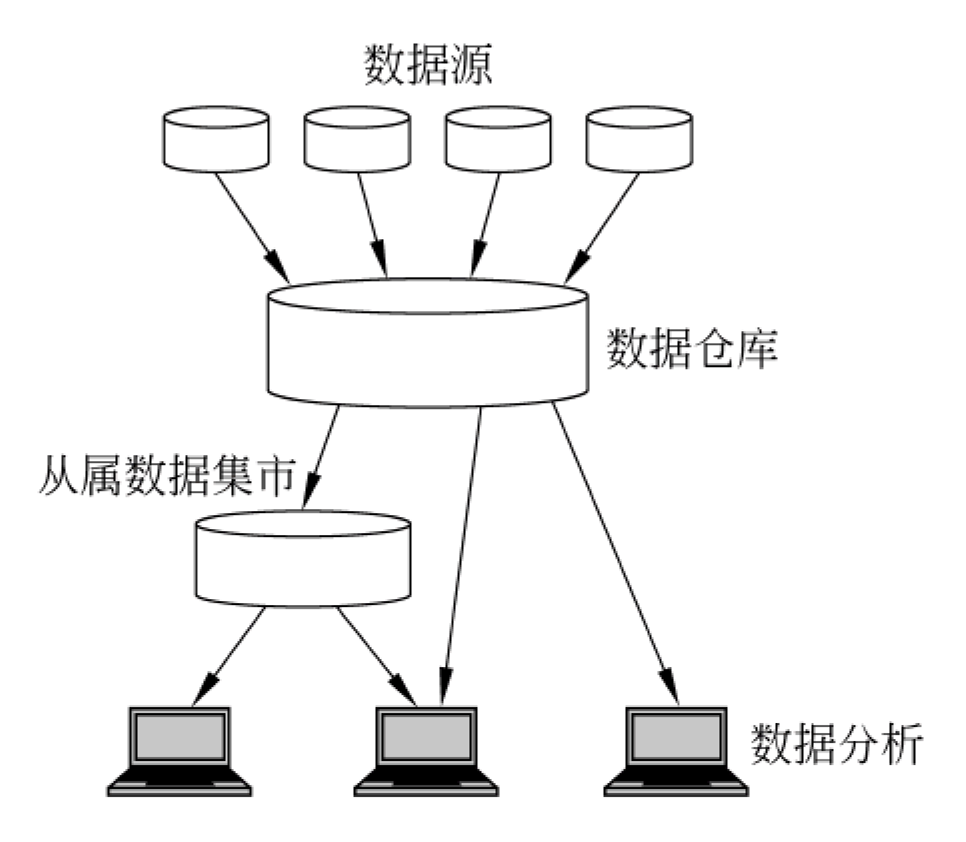


图 2.2　从属数据集市结构

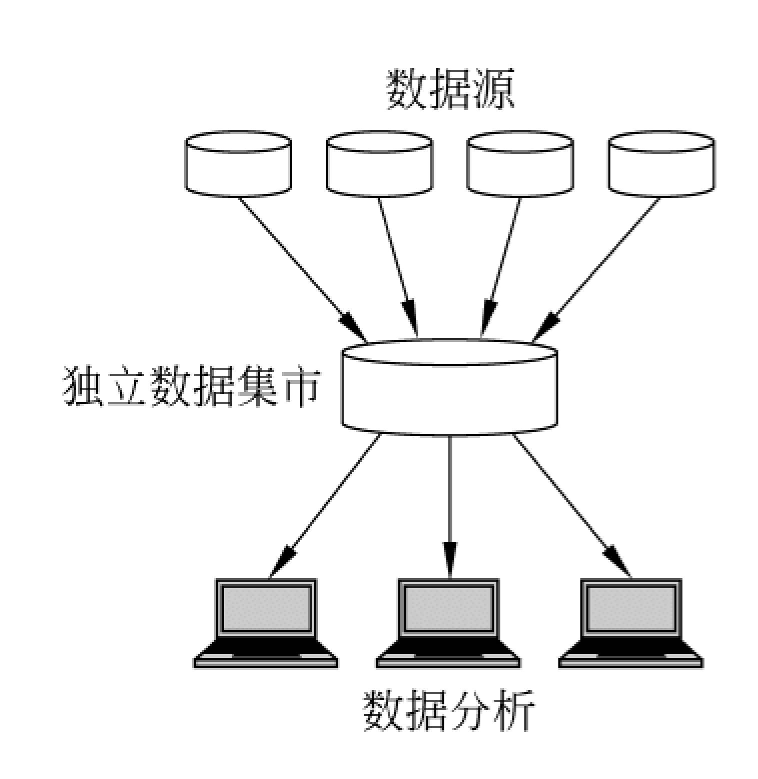


图 2.3　独立数据集市结构

独立数据集市的数据直接来源于各生产系统。许多企业在计划实施数据仓库时,往往出于投资方面的考虑,最后建成独立数据集市,用来解决个别部门比较迫切的决策问题。从这个意义上讲,它和企业数据仓库除了在数据量大小和服务对象上有所区别外,逻辑结构并无多大区别,这是把数据集市称为部门数据仓库的主要原因。

6. 关于数据集市的误区

数据集市是一个数据分支子集,可以从一个数据仓库中找到,或者是为支持一个单独业务单元的决策支持而建立的。甚至企业的大部分战略可以由数据集市来完成,在这个过程中制定行动方针。但是,在建立一个数据集市之前,企业应该知道几个关于数据集市的不切实际的看法。

(1) 单纯用数据量大小来区分数据集市和数据仓库

用大小来判断一个企业,是实施数据仓库还是数据集市的做法是很片面的。尺寸大小不是数据集市的本质特征,真正的问题在于,数据集市(它可能是一个数据仓库的子集)的数据模型一定是满足应用的特定需求的。

(2) 简单地理解数据集市容易建立

数据集市的确比数据仓库的复杂性程度低一些,因为它只针对某一需要解决的特定的商业问题,但是围绕数据获取的很多复杂问题并没有减少。

数据集市要从多个数据源中提取数据,这个过程很耗时,因为这个过程与建立一个数据

仓库一样,需要相同的计划和管理,并且需要把数据模型化。

(3) 数据集市很容易升级成数据仓库

事实上,数据集市针对特殊的业务需要,不可能很容易地伸缩。如果没有事先的扩展数据模型,追加数据是非常困难的。例如,一个数据集市可以很快找到最畅销款式的鞋的销售数字,为了增加关于这种鞋的信息,比如,新顾客的百分比,就需要新的数据模型,这种数据集市的扩充是困难的。

2.1.3　数据仓库系统结构

数据仓库系统由数据仓库、仓库管理和分析工具3部分组成。其结构形式如图2.4所示。

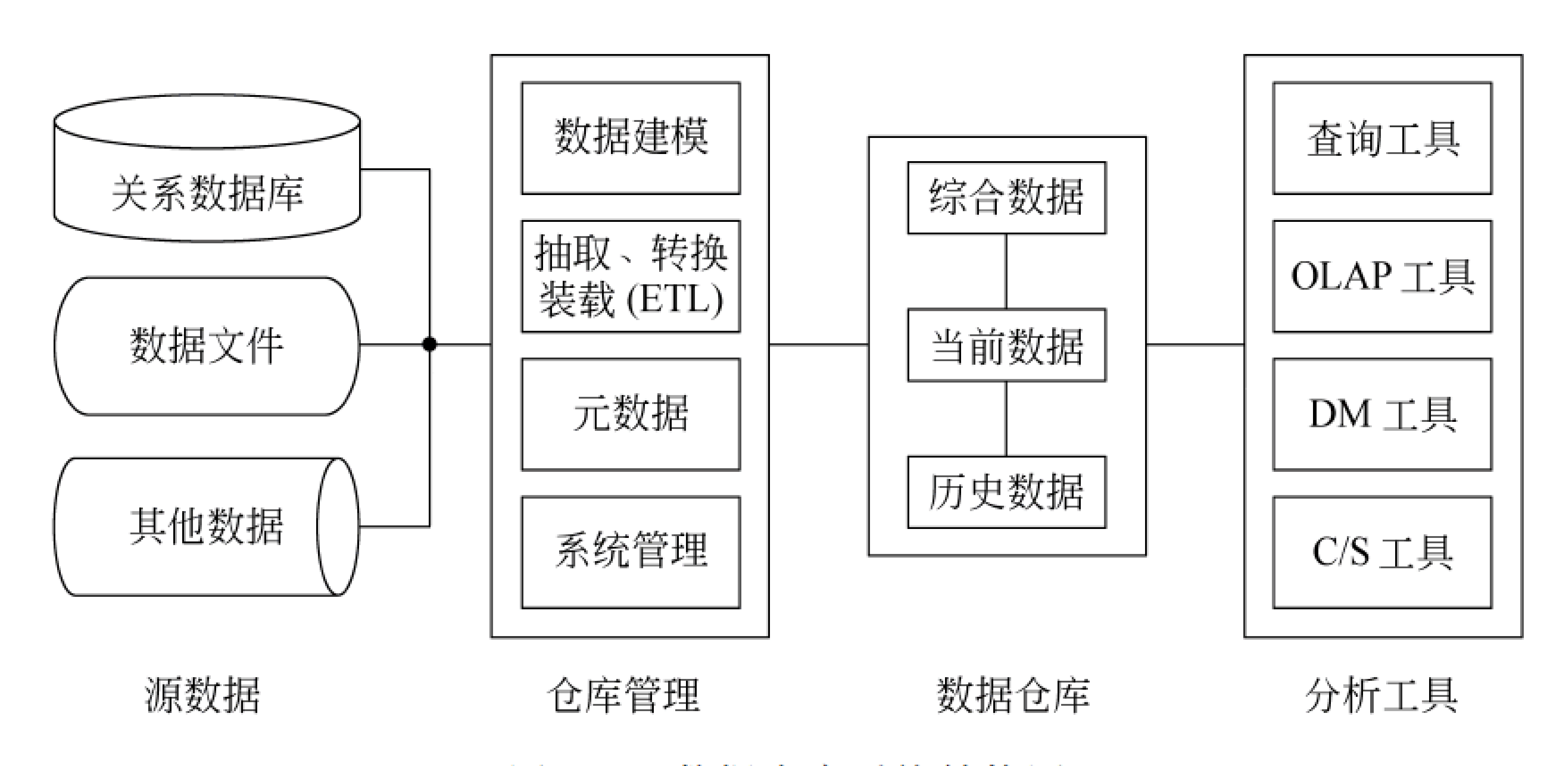


图2.4　数据仓库系统结构图

数据仓库的数据来源于多个数据源。源数据包括企业内部数据、市场调查报告以及各种文档之类的外部数据。

1. 仓库管理

仓库管理包括数据建模,数据抽取、转换、装载(ETL),元数据,系统管理4部分。

(1) 数据建模

数据建模是建立数据仓库的数据模型(data model)。数据模型是现实世界数据特征的抽象。数据模型一般包括数据结构和数据操作。数据结构包括数据类型、内容、数据之间的关系,它是数据模型的静态描述。数据操作是对数据仓库中数据所允许的操作,如检索、计算等,它是数据模型的动态描述。

数据仓库的数据模型,按数据仓库设计过程分为概念数据模型、逻辑数据模型和物理数据模型。

数据仓库的数据模型不同于数据库的数据模型在于:

① 数据仓库的数据模型的数据只为决策分析用,不包含那些纯事务处理的数据;

② 数据仓库的数据模型中增加了时间属性的代码数据;

③ 数据仓库的数据模型中增加了一些导出数据,如综合数据等。

数据仓库的数据建模是使建立的物理数据模型能适应决策用户使用的逻辑数据模型。

(2) 数据抽取、转换、装载(ETL)

数据仓库中的数据是通过在源数据中抽取数据,按数据仓库的逻辑数据模型的要求进行数据转换,再按物理数据模型的要求装载到数据仓库中去。

数据抽取、转换、装载是建立数据仓库的重要步骤,也是一项烦琐、耗时且费劲的工作,需要花费开发数据仓库70%的工作量。

(3) 元数据

元数据在数据仓库中扮演了一个新的重要角色。元数据不仅是数据仓库的字典,而且要指导数据的抽取、转换、装载工作,还要指导用户使用数据仓库。

(4) 系统管理

系统管理包括数据管理、性能监控、存储器管理以及安全管理等。

- 数据管理包括为适应竞争的变化业务需求更新数据、清理脏数据、删除休眠数据等工作。
- 系统对性能的监控是搜集和分析系统性能的信息,确定系统是否达到了所确定的服务水平。
- 存储器管理是使数据仓库的存储器要适应数据量的增长需求,实现用户的快速检索。
- 安全管理是保证应用程序的安全以及数据库访问的安全。

2. 分析工具

由于数据仓库的数据量大,必须有一套功能很强的分析工具集来实现从数据仓库中提供辅助决策的信息,完成决策支持系统的各种要求。

(1) 查询工具

数据仓库的查询不是对记录级数据的查询,而是对分析要求的查询。以图形化方式展示数据,可以帮助了解数据的结构、关系以及动态性。

(2) 多维数据分析工具(OLAP 工具)

通过对多维数据进行快速、一致和交互性的存取,这样便利用户对数据进行深入的分析和观察。

多维数据的每一维代表对数据的一个特定的观察视角,如时间、地域、业务等。

(3) 数据挖掘工具(DM 工具)

从大量数据中挖掘具有规律性的知识,需要利用数据挖掘中的各种不同算法。

(4) 客户/服务器(client/server,C/S)工具

数据仓库一般都是以服务器(server)形式在网络环境下提供服务,能对多个客户(client)同时提供服务。

2.1.4 数据仓库的运行结构

数据仓库应用是一个典型的客户/服务器(C/S)结构形式,如图 2.5 所示。数据仓库采用服务器结构,客户端所做的工作包括客户交互、格式化查询、结果显示、报表生成等。服务器端完成各种辅助决策的 SQL 查询、复杂的计算和各类综合功能等。



图 2.5　数据仓库的 C/S 结构

现在,越来越普通的一种形式是三层 C/S 结构形式,即在客户端与数据仓库服务器之间增加一个多维数据分析(OLAP)服务器,如图 2.6 所示。



图 2.6　数据仓库的三层 C/S 结构

OLAP 服务器将加强和规范化决策支持的服务工作,集中和简化了数据仓库服务器的部分工作,即 OLAP 服务器从数据仓库服务器中抽取数据,在 OLAP 服务器中转换成客户端用户要求的多维视图,并进行多维数据分析,将分析结果传送给客户端。这种结构形式工作效率更高。

2.2　数据仓库的数据模型

数据仓库不同于数据库。数据仓库的逻辑数据模型是多维结构的数据视图,也称多维数据模型,见图 2.7。

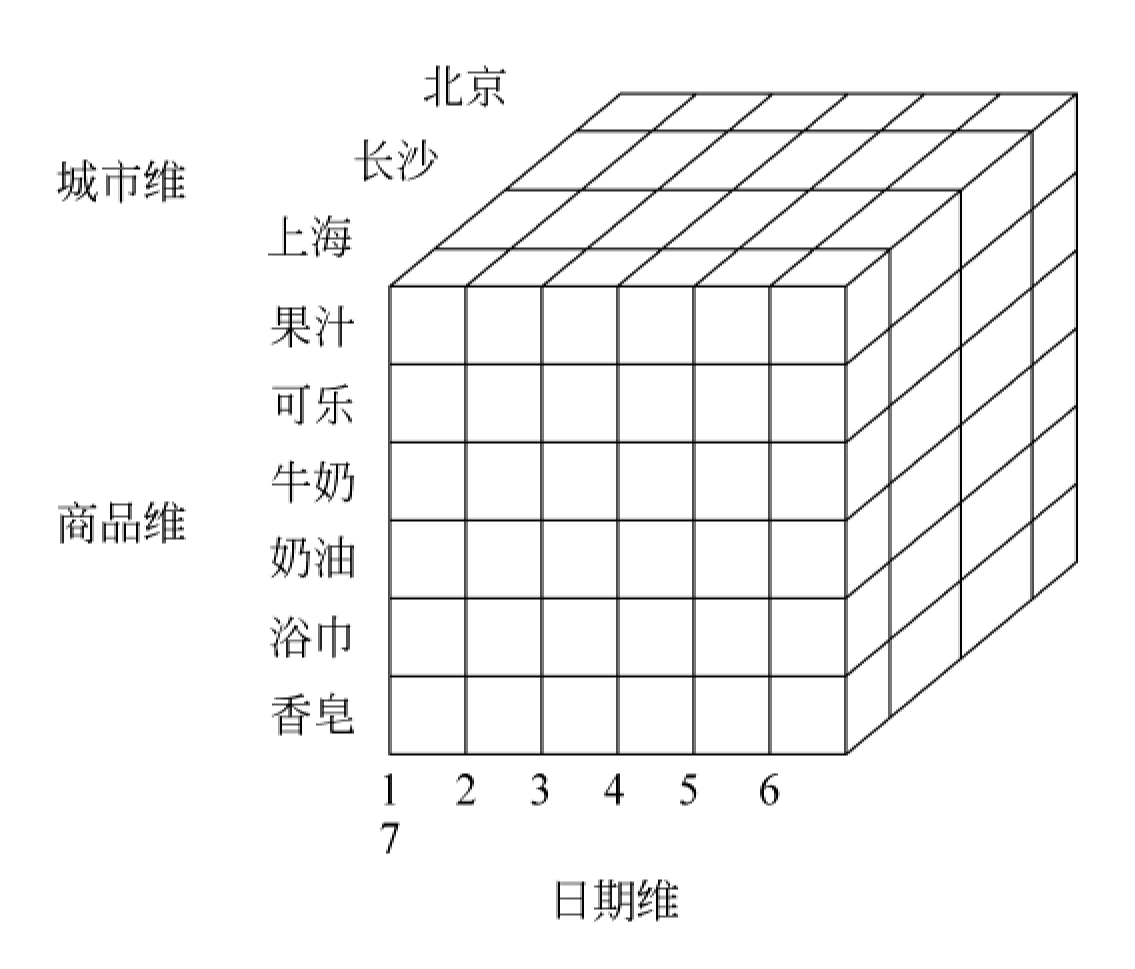


图 2.7　数据仓库的数据模型

在多维数据模型中,主要数据是数字实际值,如销售量、投资额、收入等。而这些数字实际值是依赖于一组“维”的,这些维提供了实际值的上下文关系。例如销售量与城市、商品名称、销售时间有关,这些相关的维惟一决定了这个销售实际值。因此,多维数据视图就是这些维构成的多维空间中,存放着数字实际值。图中的小格内存储的数据我们可以假设为商品的销售量。

多维数据模型的另一个特点是对一个或多个维所完成的集合运算。例如对总销售量按城市进行计算和排序。这些运算还包括对于同样维的实际值进行比较(如销售与预算)。一

般来说,时间维是一个有特殊意义的维,它对决策中趋势分析很重要。

对于逻辑数据模型,可以使用不同的存储机制和表示模式来实现多维数据模型。目前,使用的多维数据模型主要有星型模型、雪花模型、星网模型、第三范式等。

2.2.1 星型模型

大多数的数据仓库都采用“星型模型”。星型模型是由“事实表”(大表)以及多个“维表”(小表)所组成。“事实表”中存放大量关于企业的事实数据(数字实际值)。对象(元组)个数通常都很大,而且非规范化程度很高。例如,多个时期的数据可能会出现在同一个表中。“维表”中存放描述性数据,维表是围绕事实表建立的较小的表。

一个星型数据模型实例如图 2.8 所示。

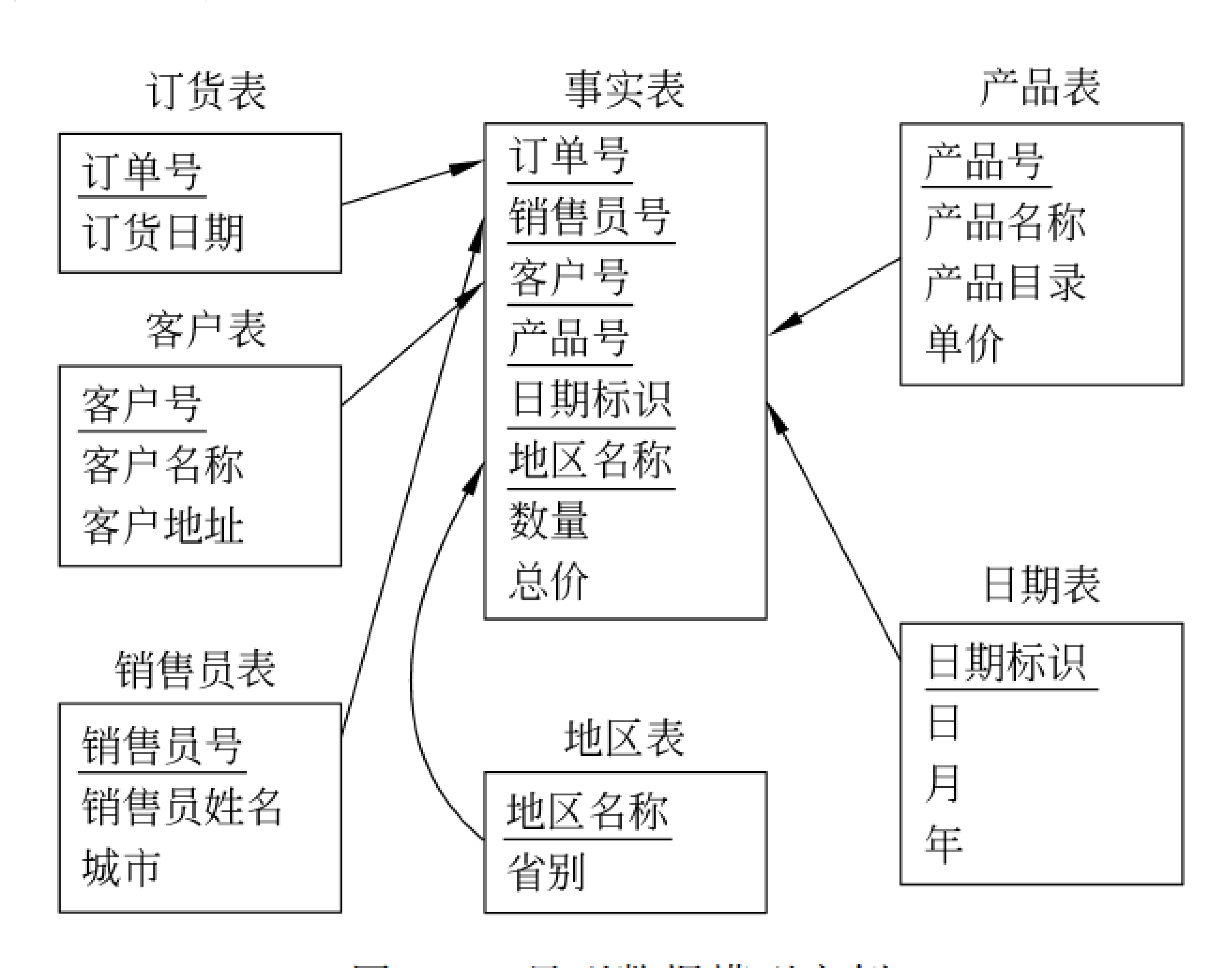


图 2.8　星型数据模型实例

事实表有大量的行(元组),然而维表相对来说有较少的行(元组)。星型模型的存储情况示意图如图 2.9 所示。

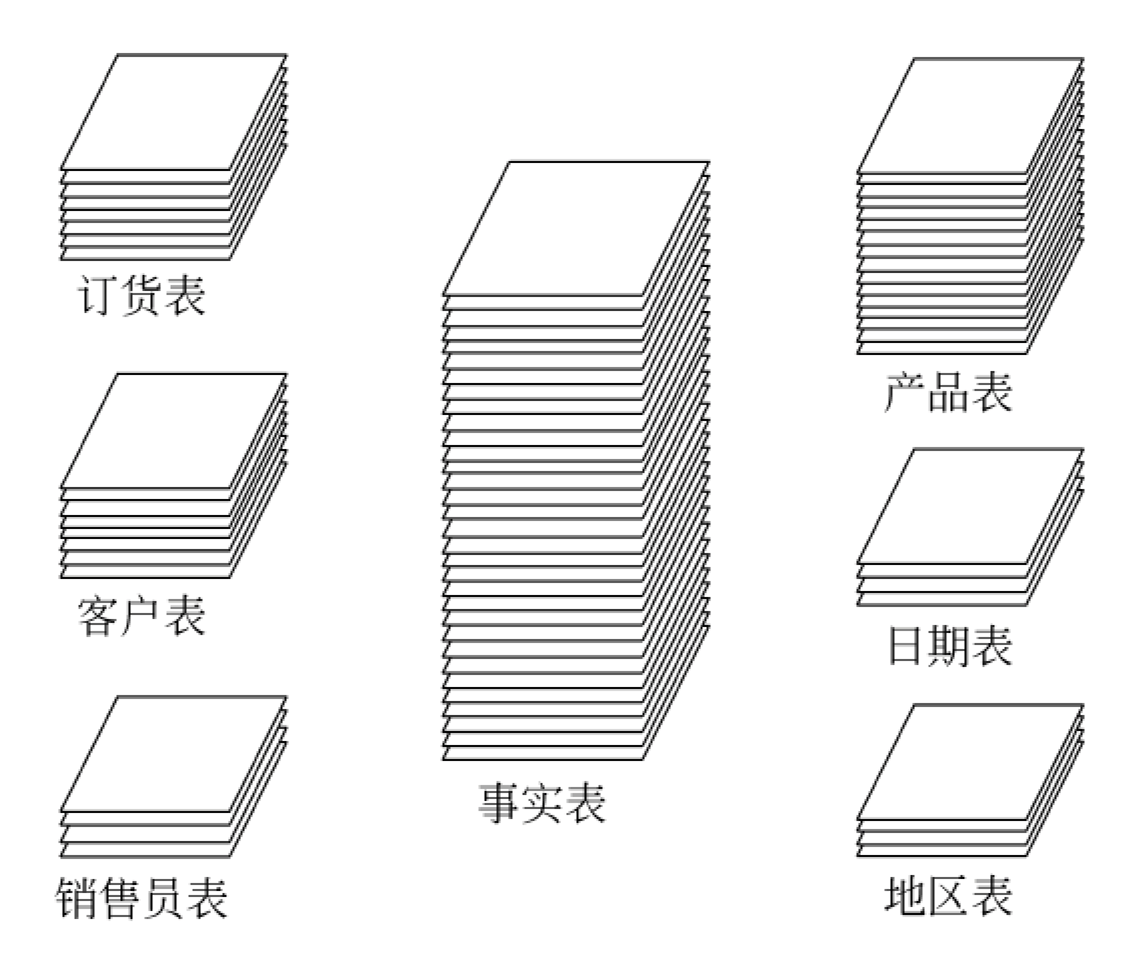


图 2.9　星型模型数据存储情况示意图

星型模型存取数据速度快,主要针对各个维做了大量的预处理,如按照维进行预先的统

计、分类、排序等；如按照汽车的型号、颜色、代理商进行预先的销售量统计，做报表时速度会很快。

星型结构与规范化的关系数据库设计相比较，存在一些显著的优点：星型模型是非规范化的，以增加存储空间代价，提高了多维数据的查询速度。而规范化的关系数据库设计是使数据的冗余保持在最少，并减少了当数据改变时系统必须执行的动作。

星型模型也有缺点：当业务问题发生变化，原来的维不能满足要求时，需要增加新的维。由于事实表的主键由所有的维表的主键组成，这种维的变化带来数据变化将是非常复杂、非常耗时的。星型模型的数据冗余量很大。

2.2.2　雪花模型

雪花模型是对星型模型的扩展，雪花模型对星型模型的维表进一步层次化，原来的各维表可能被扩展为小的事实表，形成一些局部的"层次"区域。它的优点是最大限度地减少数据存储量，以及把较小的维表联合在一起来改善查询性能。

雪花模型增加了用户必须处理的表的数量，增加了某些查询的复杂性。但这种方式可以使系统更进一步专业化和实用化，同时降低了系统的通用程度。前端工具将用户的需求转换为雪花模型的物理模式，完成对数据的查询。

在雪花模型中能够定义多重"父类"维来描述某些特殊的维表。比如，在时间维上增加了月维和年维，通过查看与时间有关的父类维，能够定义特殊的时间统计信息，如销售月统计、销售年统计等。

在图 2.8 星型模型的数据中，对"产品表"、"日期表"、"地区表"进行扩展形成雪花模型数据，如图 2.10 所示。使用数据仓库的工具完成一些简单的二维或三维查询，既满足了用户对复杂的数据仓库查询的需求，又能够完成一些简单查询功能而不用访问过多的数据。

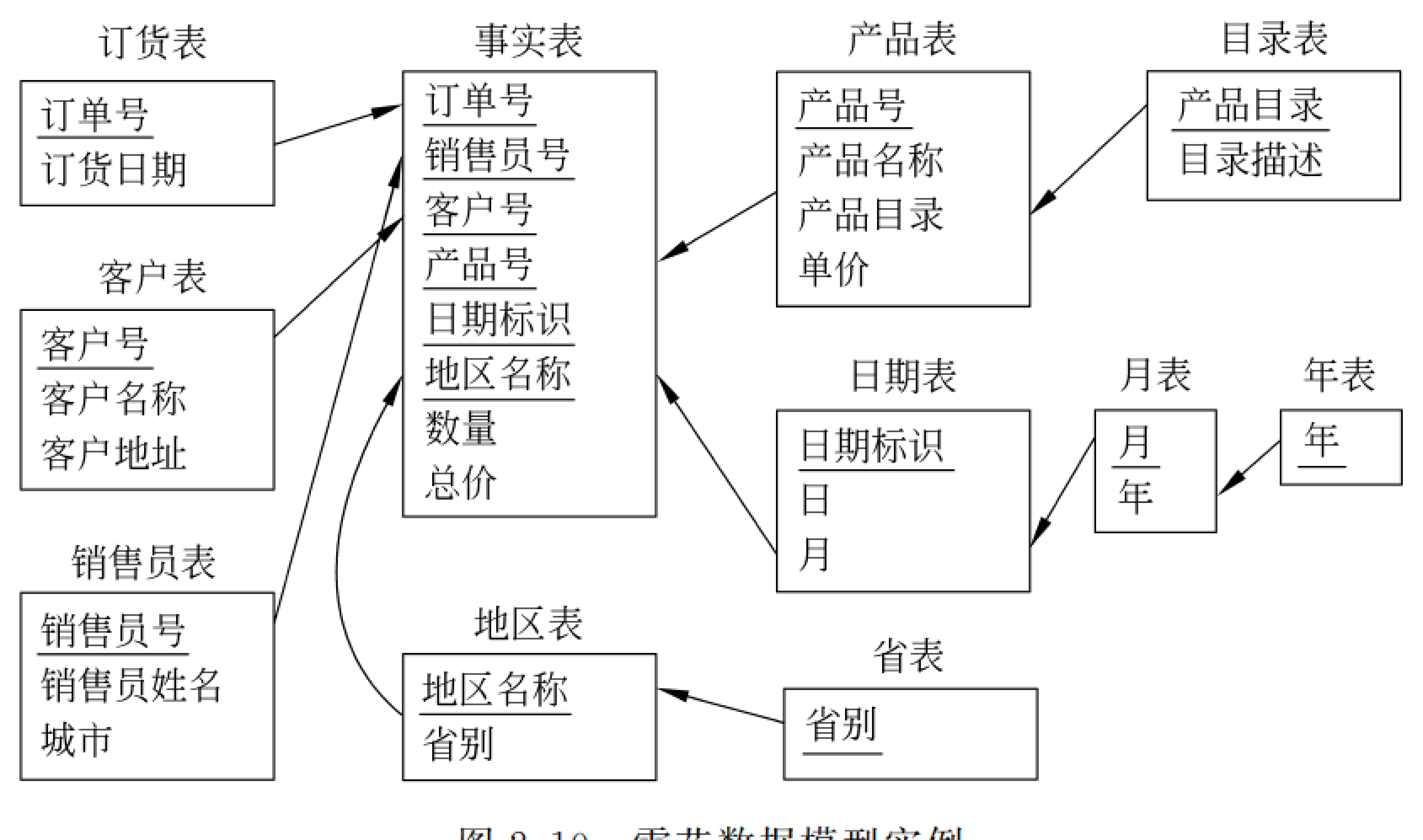


图 2.10　雪花数据模型实例

2.2.3　星网模型

每个数据仓库都包含了多个星型模型的结构。每一个星型模型都在事实表中保存了一

些指标，为特定的目的服务。多个相关的星型模型通过相同的维表连接起来形成网状结构，称为星网模型。在大多数星网模型中，各个事实表共享的维表是时间维。

构造星型模型有几种情况：有的是增加汇总事实表和衍生的维表形成星网模型；有的是构造相关的事实表形成星网模型。

例如，电话公司需要建立两个事实表，一个事实表跟踪单独的电话事务，它能回答“节假日电话收益与工作日电话收益的对比情况”等类问题；另一个事实表累计用户电话支出情况，它能回答“某个用户在某段时间内的电话余额”等类问题。该电话公司星网模型实例如图 2.11 所示。

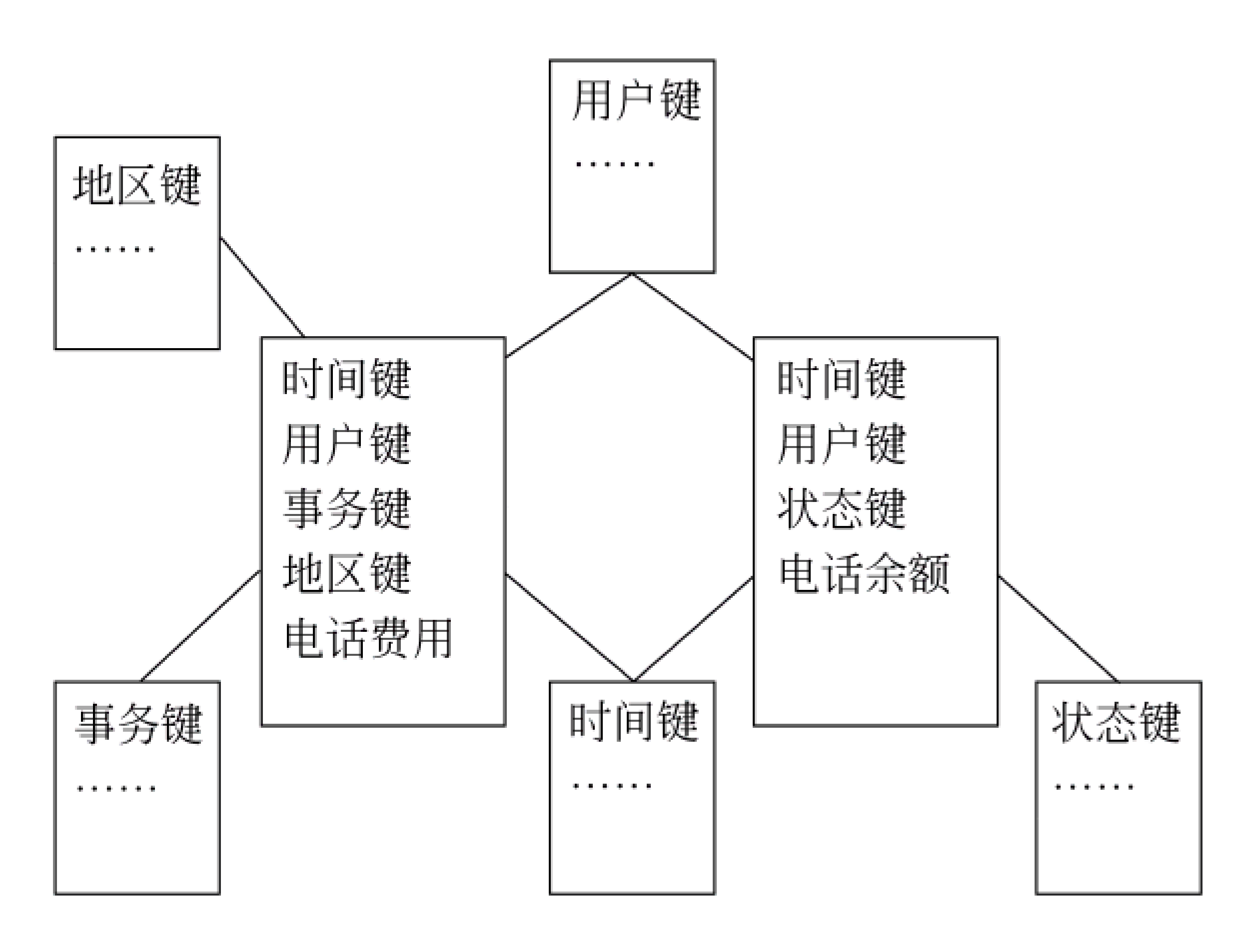


图 2.11　电话公司星网模型实例

2.2.4　第三范式

范式实际上是传统的关系数据库的设计理论。一个规范化的关系模式应该准确地反映所描述的数据实体，避免冗余、异常（插入异常、删除异常、更新异常）等问题。

通常按照属性间依赖情况来区分关系规范化的程度，现有第一范式到第五范式。

数据仓库可以按第三范式进行逻辑数据建模。它不同于星型模型，是把事实表和维表的属性作为一个实体都集中在同一数据库表中，或分成多个实体用多个表来表示，表按第三范式组织数据，减少了维表中的键和不必要的属性。

著名的 NCR 数据仓库公司采用了第三范式的逻辑数据模型。现在有很多大型的企业数据仓库系统中都同时采用了第三范式和星型模型，即用第三范式来描述数据仓库系统后台的详细数据存储关系，在此基础上，再根据特定的分析需求建立适当的星型模型，用于刷新 OLAP 服务器的立方体（cube），以方便前端数据展现和预定义的多维分析。

星型模型在进行多维数据分析时，在不超过预定义的维度范围内，速度是很快的，但是在超出了预定义的维度，增加维度将是很困难的事情。

第三范式对于海量数据（如 TB 级），且需要处理大量的动态业务分析时，就显示了它的优势。

2.3 数据抽取、转换和装载

数据仓库的数据来源于多个数据源，主要是企业内部数据(用于企业的事务处理，也称操作型数据)、存档的历史数据、企业的外部数据(本行业的统计数据以及竞争者的市场占有率数据等)。这些数据源可能是在不同的硬件平台上，使用不同的操作系统。源数据是以不同的格式存放在不同的数据库中。

数据仓库需要将这些源数据经过抽取、转换和装载的过程，存储到数据仓库的数据模型中。可以说，数据仓库的数据获取需要经过抽取(extraction)、转换(transform)和装载(load)3 个过程，即 ETL 过程。

经过 ETL 过程，将源系统中的数据改造成有用的信息，存储到数据仓库中。例如，ETL 过程将统一各源系统中数据的变量名称，转换和集成所有产品的销售情况数据，装载到数据仓库的销售事实表和相关维表中。在用户查询时，在事实表中提供销售数量与金额的同时，在产品维度表中提供产品目录，在商店维度中提供商店名单，在时间维度中提供日期。这种查询便利情况对比和决策分析。

ETL 过程在开发数据仓库时，占去 70%的工作量。ETL 过程的主要步骤概括为：

(1) 决定数据仓库中需要的所有的目标数据；

(2) 决定所有的数据源，包括内部和外部的数据源；

(3) 准备从源数据到目标数据的数据映射关系；

(4) 建立全面的数据抽取规则；

(5) 决定数据转换和清洗规则；

(6) 为综合表制定计划；

(7) 组织数据缓冲区域和检测工具；

(8) 为所有的数据装载编写规程；

(9) 维度表的抽取、转换和装载；

(10) 事实表的抽取、转换和装载。

2.3.1 数据抽取

数据抽取工作包括以下两点。

1. 确认数据源

对数据源的确认不仅是对数据源的简单确认，还包括检查和确定数据源是否可以提供数据仓库需要的数据。该项工作包括：

(1) 列出对事实表的每一个数据项和事实；

(2) 列出每一个维度属性；

(3) 对于每个目标数据项，找出源数据项；

(4) DW 中一个数据元素有多个来源，选择最好的来源；

(5) 确认一个目标字段的多个源字段，建立合并规则；

(6) 确认多个目标字段的一个源字段,建立分离规则;

(7) 确定默认值;

(8) 检查缺失值的源数据。

2. 数据抽取技术

数据抽取时要考虑两种情况:

(1) 当前值。源系统中存储的数据都代表了当前时刻的值,当商业交易时,这些数据是会发生变化的。

(2) 周期性的状态。这类数据存储的是每次发生变化时的状态。例如,对于每一保险索赔,都经过索赔开始、确认、评估和解决等步骤,都要考虑时间说明。

在建立数据仓库时,从某一特定时间开始的最初数据必须迁移到数据仓库中,以使数据仓库开始运转,这是初始装载。在初始装载之后,数据仓库必须保持更新,使变化的历史和状态可以在数据仓库中反映出来。

数据抽取完成两类数据的抽取:

(1) 静态数据的抽取。一般在数据仓库的初始装载时抽取的是静态数据,它代表了某个时刻的快照。

(2) 修正数据的抽取。它也称为追加的数据抽取。修正数据的抽取过程包括特定时刻抽取的数据值,分为立即型数据抽取(实时的数据抽取)和延缓型的数据抽取。

立即型数据抽取的典型方法是通过读取交易日志抽取所有相关交易记录。一般利用复制技术从交易日志中捕获交易日志中的变化数据,从日志传输到目标文件中,并检验数据变化的传输情况,确保复制的成功。

延缓型数据抽取的典型方法是,通过读取源记录中包括日期和时间的标记,抽取更新源记录的数据。如果没有时间标记的旧数据源,就要通过"快照对比技术",即通过比较源数据的两个快照来抽取变化的数据。

2.3.2 数据转换

数据抽取过程中得到的数据是没有经过加工的数据,不能直接应用于数据仓库,必须经过多种处理,将抽取的数据转换成可以存储在数据仓库中的信息。

1. 数据转换的基本功能

(1) 选择。从源系统中选择整个记录或者部分记录。

(2) 分离/合并。对源系统中记录中的数据进行分离操作或者对很多源系统中选择的部分数据进行合并操作。

(3) 转化。对字段的转化包括对源系统进行标准化和使字段对用户来说是可用和可理解的。

(4) 汇总。数据仓库中需要保存很多汇总数据。这需要将最低粒度数据进行汇总。例如,对零售连锁店需要将每一个收款机的每一笔交易的销售数据,汇总为每天每个商店关于每种商品的数据。

(5) 清晰化。对单个字段数据进行重新分配和简化,使数据仓库更便利使用。

2. 数据转换类型

(1) 格式修正。包括数据类型和单个字段长度的变化,例如在源系统中,产品类型通过代码和名称在数值型和文本类型中表示,不同的源系统将会有所不同,对这些数据类型进行标准化,改变成更有意义的文本值。

(2) 字段的解码。对所有晦涩的编码进行解码,将它们变成用户可以理解的值。例如,对性别的解码,在源系统中有的用 1 和 2 表示,有的用 M 和 F 表示男性和女性。

(3) 计算值和导出值。在数据仓库中,有时需要用销售和成本一起计算出利润值。导出字段包括平均每天的收支差额和相关比率。

(4) 单个字段的分离。在旧系统中将客户名称、地址存放在大型文本字段中;姓和名存放在一个字段中;城市、地区和邮政编码存放在一个字段中。在数据仓库中却需要将姓名和地址存放在不同的字段中,便利不同要求的分析工作。

(5) 信息的合并。例如,一个产品的信息可能从不同的数据源中获得:产品编码和产品名从一个数据源得到;相关包装类型从另一个数据源中得到;成本数据从第三个数据源中得到。信息合并是将产品编码、产品名、包装类型和成本的有机组合,成为一个新的实体。

(6) 特征集合转化。例如,在源系统中数据采用 EBCDIC 码,而数据仓库数据采用 ASCII 码,这将要进行代码集合的转化。

(7) 度量单位的转化。使数据具有相同的标准度量单位。不少国家有自己的度量单位,需要在数据仓库中采用标准度量单位。

(8) 日期/时间转化。日期和时间的表示应该转化成国际标准格式。如 2005 年 10 月 15 日在美国表示成 10/15/2005,而在英国表示为 15/10/2005。标准格式为 15 OCT 2005。

(9) 汇总。这种类型的转换是创建数据仓库的汇总数据。汇总数据适合于客观战略性的查询。

(10) 关键字重新构造。在源系统中关键字可能包含很多项的内容。如产品编码包括仓库代码、销售区域、产品编码等多项内容。在数据仓库中,关键字要发生变化,转换成适合于事实表和维表的普通键值。

3. 数据整合和合并

数据仓库的数据是从很多不同的分散的源系统中的源数据集成起来的。各源系统采用不同的命名方式和不同的数据标准。数据整合和合并是将相关的源数据组合成一致的数据结构,装入数据仓库。具体表现如下:

(1) 实体识别问题

例如,一个数据仓库的数据来源于 3 个不同的客户系统。一个系统是订单登记系统,一个是客户服务支持系统,一个是市场系统。这 3 个系统中对相同客户可能分别有不同的键码。

在数据仓库中,需要为每一个客户建立一个记录,就必须从 3 个源系统中得到同一客户的数据,将它们组合成一条单独的记录。这是客户实体识别问题。

进行数据转换时，需要让用户参与这个过程，帮助对实体的识别，并设计算法，将 3 个系统中得到的记录进行匹配，建立统一的记录集合。

(2) 多数据源相同属性不同值的问题

例如，假设产品的单位成本可能从两个系统中得到，在特定的时间间隔内对成本值进行计算和刷新，由于两个系统中得到的成本存在一些差别，数据仓库应该从哪个系统中取得成本呢？

有 3 种方法：

① 分别给这两个系统不同的优先权，取高优先权的成本数据；

② 根据最新的刷新日期来选择其中一个源系统的成本数据；

③ 根据其他相关字段来选择合适的源系统的成本数据。

4. 如何实施转换

完成数据转换工作一般采用两种方式：自己编写程序实现数据转换和使用转换工具。

(1) 自己编写程序实现数据转换

在明确了数据转换的类型和数据整合与合并的内容后，一般具有编程能力的程序员和分析师都可以编写数据转换程序。这种方式会带来复杂的编程和测试。

(2) 使用转换工具

使用自动的工具会提高效率和准确性。当确定数据转换参数和规则时，将它作为元数据存储在工具中，工具就能有效地完成数据转换工作。这是使用数据转换工具的主要优点。

2.3.3 数据装载

一旦创建了装载映像，数据转换功能就结束了，接下来的是数据装载。它完成将转换好的数据存储到数据仓库的数据库中去。

数据装载工作包括数据装载方式和数据装载类型。

1. 数据装载方式

(1) 基本装载

按照装载的目标表，将转换过的数据输入到目标表中去。若目标表中已有数据，装载时会先清除这些数据，再装入新数据。目标表可以是事实表或维表。

(2) 追加

如果目标表中已经存在数据，追加过程在保存已有数据的基础上增加输入数据。当一个输入数据记录与已经存在的记录重复时，输入记录可能作为副本增加进去，或者丢弃新输入的数据。

(3) 破坏性合并

如果输入数据记录的主键与一条已经存在的记录的键互相匹配时，用新输入数据更新目标记录数据。如果输入记录是一条新的记录，没有任何与之匹配的现存记录，那么就将这条输入记录添加到目标表中。

(4) 建设性合并

如果输入记录主键与已有记录的键相匹配时,保留已有的记录,增加输入的记录,并标记为旧记录的替代。

2. 数据装载类型

数据装载类型包括 3 种:最初装载、增量装载和完全刷新。

(1) 最初装载

这是第一次对整个数据仓库进行装载。在装载工作完成以后建立索引,这样可以减少创建索引时间。

(2) 增量装载

由于源系统的变化,数据仓库需要装载变化的数据,这就是增量装载。

在建设性合并的装载方式中,对增加的输入记录中标记了旧记录的替代,这可以作为增量装载的方法。

对于已装入的记录数据必须被改正后的数据记录取代时,要采用破坏性合并的装载方式作为增量装载的方法。

(3) 完全刷新

这种类型的数据装载用于周期性重写数据仓库。有时,也可能对一些特定的表进行刷新。完成刷新与初始装载比较相似。不同点在于在完全刷新之前,目标表中已经存在数据。初始装载和追加装载都可以应用于完全刷新中。

2.3.4 ETL 工具

目前市场上有 3 类 ETL(数据抽取、转换、装载)工具。

1. 数据转换引擎

这类工具根据用户定义的时间间隔,从一组指定的源系统中抽取数据,执行复杂的数据转换,将结果导入到目标表中。使用这类工具可以选择最合适的数据转换方法,实施完全更新和增量装载。

这类工具的功能涵盖了整个 ETL 过程。

2. 通过复制捕获数据

这类工具中大部分使用由数据库管理系统维护的交易日志。在交易日志中捕获的源系统的变化,可以近乎实时地在数据准备区域被复制,等待进一步的处理。

3. 代码生成器

这类工具根据提供的数据源的参数和目标输出以及商业规则,能自动生成数据抽取和转换程序,完成 ETL 过程。

这类工具的自动化程度较高。

对数据仓库的数据抽取、数据转换和数据装载过程,选择 ETL 工具时,需要考虑以下

特征：

(1) 从多种关系型数据库中抽取数据；

(2) 从旧数据库、索引文件和平面文件中抽取数据；

(3) 源字段和目标字段从一种格式向另一种格式进行数据转换；

(4) 执行标准转化、重定义键和结构性变化；

(5) 提供从数据源到目标的检查轨迹；

(6) 抽取和转换中商业规则的应用；

(7) 将源系统中的几个记录组合成一个整合的目标记录；

(8) 元数据的记录和管理。

2.4 元　数　据

2.4.1 元数据的重要性

元数据在数据仓库的建造、运行中有着极其重要的作用。元数据描述了数据仓库的数据和环境，遍及数据仓库的所有方面，是整个数据仓库的核心。

元数据可分为4类，分别为关于数据源的元数据、关于数据模型的元数据、关于数据仓库映射的元数据和关于数据仓库使用的元数据。

下面是元数据的一个例子，它定义了数据仓库中的一个表，如表2.1所示。

表2.1　元数据举例

Table	逻辑名	顾　客
	定义	购买商品的个人或组织
	物理存储	DB.table(数据库表)
	建立日期	2003年1月15日
	最后更新日期	2005年1月20日
	更新周期	每月
	表编辑程序名	ABC(程序名)

最基本的元数据相当于数据库系统中的数据字典。由于数据仓库与数据库有很大的不同，因此元数据的作用远不是数据字典所能相比的。元数据在数据仓库中有着举足轻重的作用，它不仅定义了数据仓库有什么，指明了数据仓库中数据的内容和位置，刻画了数据的抽取和转换规则，存储了与数据仓库主题有关的各种商业信息，而且整个数据仓库的运行都是基于元数据的，如数据的修改、跟踪、抽取、装入、综合等。

有两类人会用到元数据：最终用户(包括商业分析员)和IT人员(包括开发人员和管理人员)。

1. 最终用户

数据仓库的用户希望从数据仓库获取信息回答以下问题：

- 每个商店各种产品每天的销售数量和金额是按照每一笔交易，还是按照汇总数据

存储？
- 销售情况能够按照产品、促销、商店和月份进行分析吗？
- 当月的销售能与去年同期销售对比吗？
- 销售情况能与预期目标进行比较吗？
- 利润率是如何计算的？商业规则有哪些？
- 销售区域是如何划定的？需要分析的两个区域包含了哪些地区？
- 销售情况的数据从何而来？来自哪些源系统？
- 销售数据是什么时候的？这些数据多久更新一次？

最终用户需要的元数据包括：数据内容、汇总数据、商业维度、商业指标、浏览路径、源系统、外部数据、数据转换规则、最后更新日期、数据装载和更新周期、查询模板、报表格式、预定义查询和报表、OLAP 数据等。

最终用户需要的元数据也称为商业元数据，它像一幅公路地图，显示了信息所在的地方，以及如何到达那个地方。最终用户通过商业元数据的引导，能够有效地从数据仓库中获得所需要的信息，提高分析效果。

2. IT 人员

元数据对数据仓库的开发者和管理者来说都很重要。从开始的数据抽取、数据转换、数据集成、数据清洗、数据准备、数据存储，到查询及报表设计、OLAP 设计以及运行时的管理工作，IT 人员必须能够得到合适的元数据。

IT 人员需要的元数据包括：源数据结构、源平台、数据抽取方法、外部数据、数据转换规则、数据清洗规则、准备区域结构、维度模型、初始装载、增量装载、数据汇总、OLAP 系统、Web 访问、查询和报表设计。

IT 人员需要的元数据也称为技术元数据，为负责开发、管理和维护数据仓库服务。技术元数据对 IT 人员来说，就像一个支持技术工作的指南。

2.4.2 关于数据源的元数据

这类元数据是现有业务系统的数据源的描述信息，是对不同平台上的数据源的物理结构和含义的描述。具体为：

(1) 数据源中所有物理数据结构，包括所有的数据项及数据类型；

(2) 所有数据项的业务定义；

(3) 每个数据项更新的频率，以及由谁或哪个过程更新的说明；

(4) 每个数据项的有效值；

(5) 其他系统中具有相同业务含义的数据项的清单。

2.4.3 关于数据模型的元数据

这类元数据描述了数据仓库中有什么数据以及数据之间的关系，是用户使用管理数据仓库的基础。这种元数据可以支持用户从数据仓库中获取数据。用户可以提出需要哪些表，系统从中选一个表，并得到表之间的关系。重复该过程，用户能够得到希望的数据。

为了描述数据仓库中的数据及数据之间的各种复杂关系，元数据要定义以下内容：

(1) I/O 对象：支持数据仓库 I/O 操作的各种对象。元数据要描述该 I/O 对象的定义、类型、状态、存档(刷新)周期。

(2) 关系：两个 I/O 对象之间关联。这种关联有 3 种类型：一对一、一对多和多对多。

(3) 关系成员：描述每个关系中两个 I/O 对象的具体角色(在一对多中是父亲还是儿子)、关系度(一对一还是一对多)及约束条件(必须满足还是可选关系)。

(4) 关系关键字：描述两个 I/O 对象是如何建立关联的。每个关系都是通过 I/O 对象的关键字来建立的，元数据要指明建立每个关系的相应对象的关键字。

这组元数据定义的数据之间的关系可以用图 2.12 来表示。

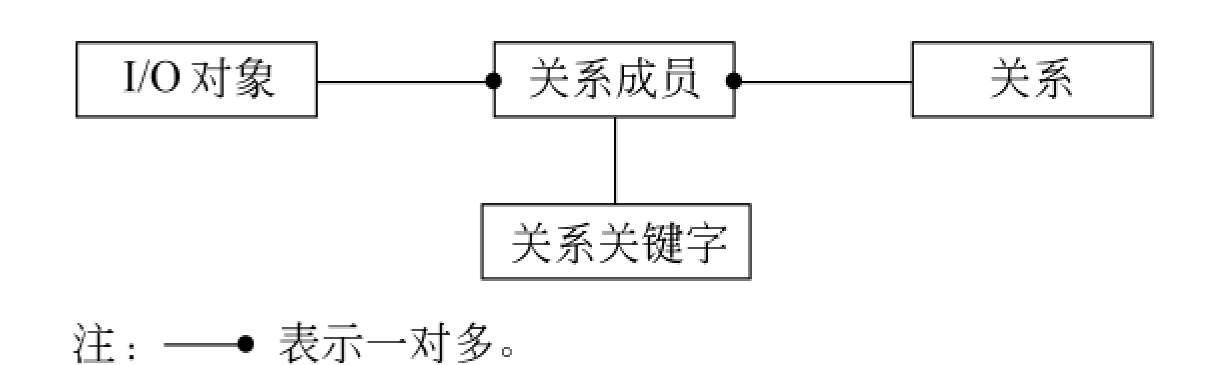


图 2.12　数据模型的元数据内容

例如，雇员与技能之间的关系如图 2.13 表示。

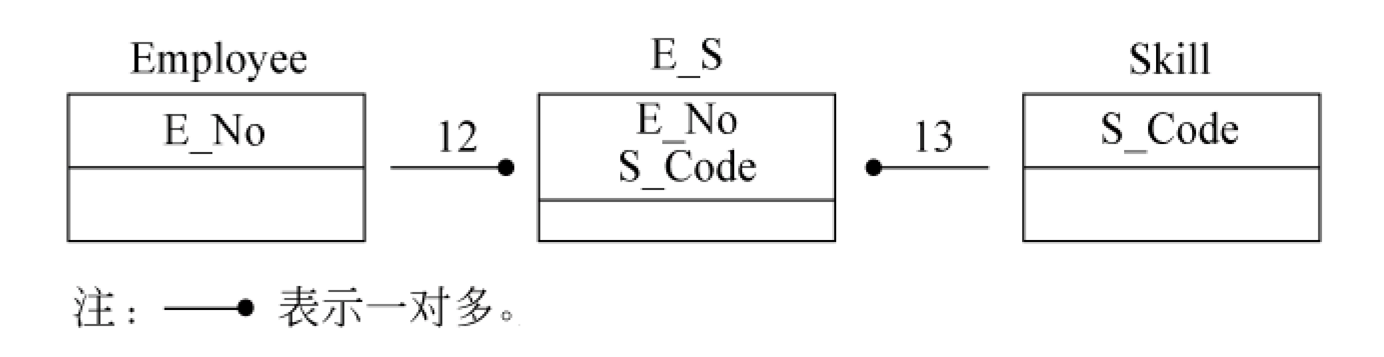


图 2.13　雇员与技能之间的关系图

在数据仓库中元数据描述该关系如图 2.14 所示。

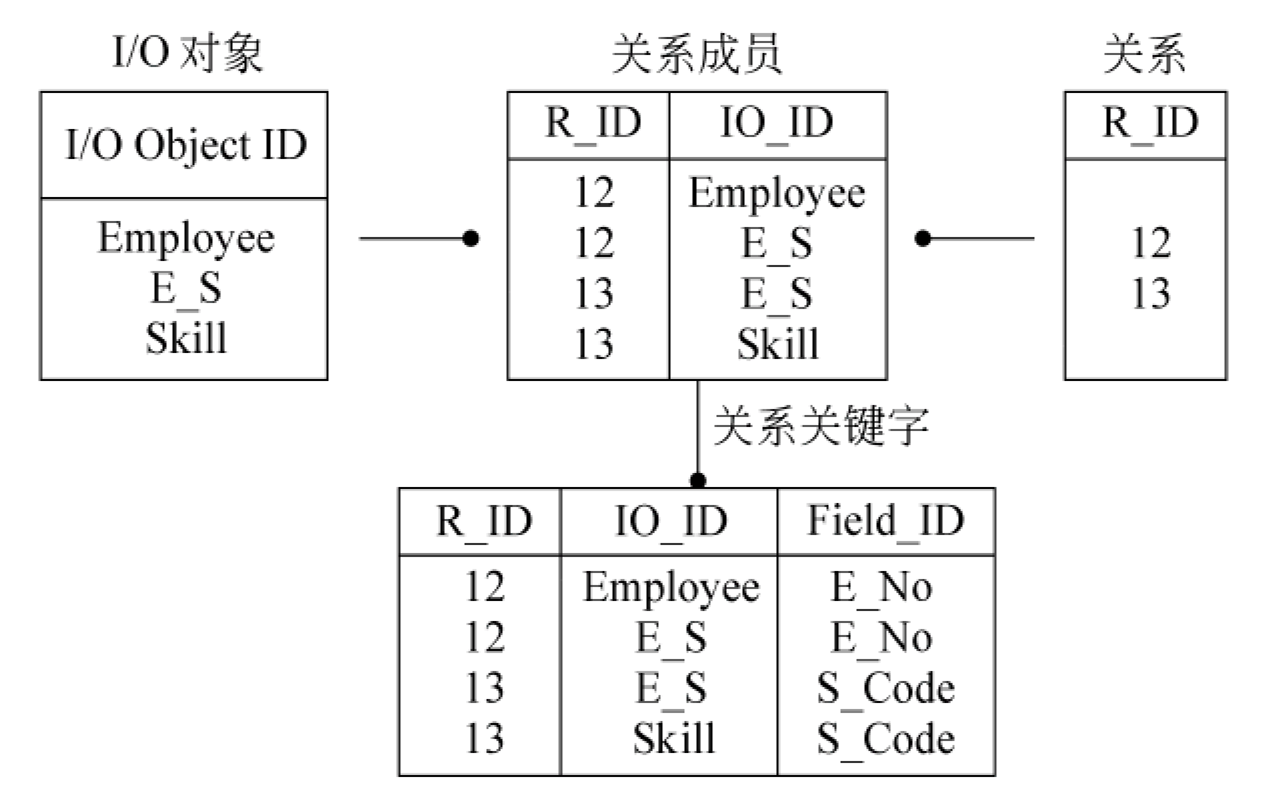


图 2.14　雇员与技能关系的元数据内容

2.4.4　关于数据仓库映射的元数据

这类元数据是数据源与数据仓库数据之间的映射。

当数据源中的一个数据项与数据仓库建立了映射关系，就应该记下这些数据项发生的任何变换或变动，即用元数据反映数据仓库中的数据项是从哪个特定的数据源抽取的，经过哪些转换、变换和装载过程。

从源系统的数据到数据仓库中的目标数据的转移是一项复杂的工作，其工作量占整个数据仓库开发的70%。这里主要涉及两个问题：

1. 抽取工作之间的复杂关系

一个数据的抽取要经过许多步骤，如图2.15所示。

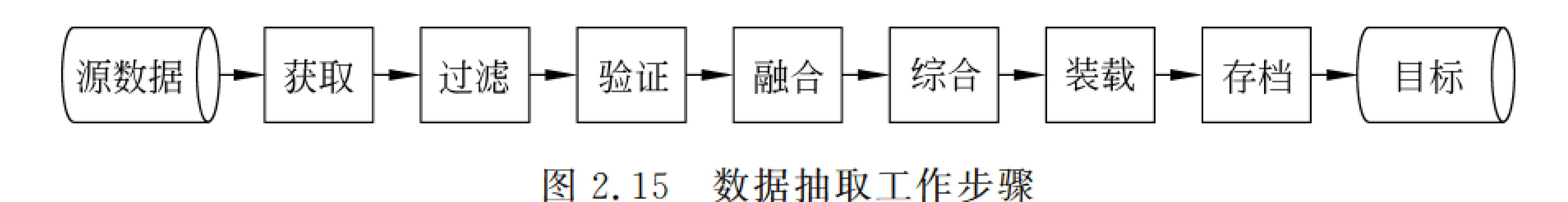


图2.15　数据抽取工作步骤

(1) 获取：从外部或内部源数据系统获取对决策支持系统用户有用的数据。

(2) 过滤：过滤掉不需要的内容(如上次抽取后一直没改变的数据)。

(3) 验证：从用户的角度验证数据的质量。

(4) 融合：把本次抽取的数据与数据仓库中的数据进行融合。

(5) 综合：对数据进行综合，生成综合级数据。

(6) 装载：把新数据装入到数据仓库中。

(7) 存档：把新装入的数据单独存为一个文件，以便减少更新操作的数据量。

2. 源数据与目标数据之间的映射

源数据与目标数据之间是一种复杂得多对多关系。元数据要能够描述这些限制所带来的一系列问题。这组元数据要定义的内容有：

(1) 抽取工作：描述每一个抽取工作，并为每一个抽取工作标识其源系统，明确其刷新周期(两次抽取之间的间隔)。

(2) 抽取工作步骤：定义抽取工作中的步骤，包括说明每一步的类型(如过滤、验证等)。

(3) 抽取表映射：为每一个抽取步骤建立输入文件/表与输出文件/表之间的关联。

(4) 抽取属性映射：为每一个抽取步骤建立输入表(文件)的属性与输出表(文件)的属性之间的关联。

(5) 记录筛选规则：在抽取工作的每一步骤中进行记录的筛选。如例子：

```
IF Record.Last_Update_Date>'2003_11_01'OR Record.Create_Date > '2003_11_01'
THEN Reserve(保留) ELSE Delete(删除)
```

这类元数据要定义的数据之间的关系表示如图2.16所示。

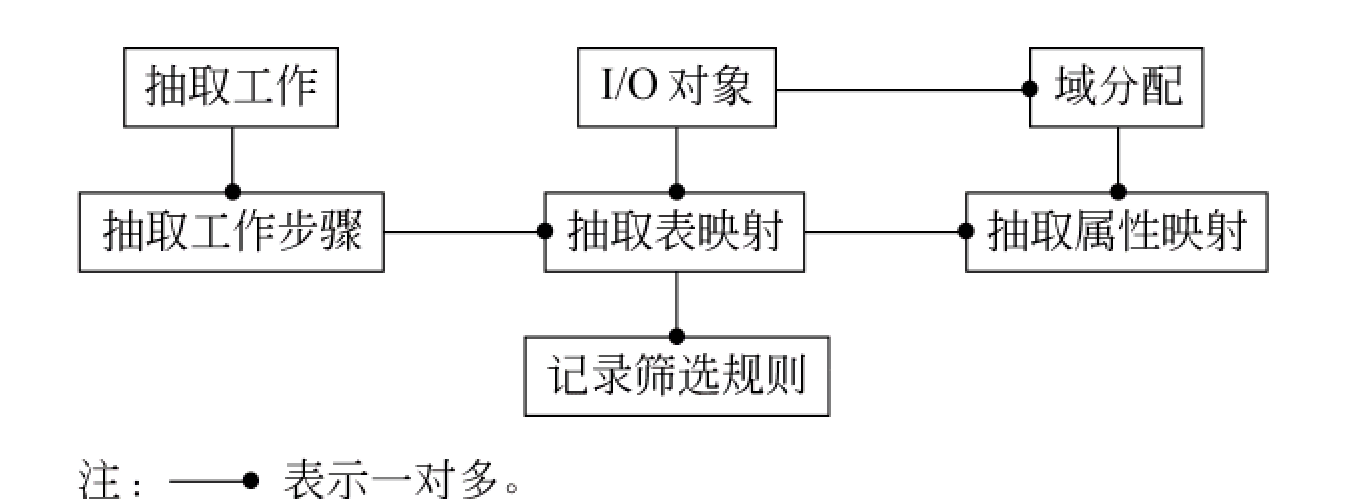


图 2.16　数据映射的元数据内容

这类元数据可以用来生成源代码，以完成数据的转换工作，即完成由操作型数据转换成面向主题的数据仓库的数据。元数据中的抽取表映射和抽取属性映射定义了进行实际抽取转换工作的过程。数据仓库管理核心利用该类元数据所定义的抽取过程生成某种语言的源代码（如 VC），然后编译成可执行的程序，以完成数据的抽取工作。

2.4.5　关于数据仓库使用的元数据

这类元数据是数据仓库中信息的使用情况描述。

数据仓库的用户最关心的是两类元数据：

（1）元数据告诉数据仓库中有什么数据，它们从哪里来，即如何按主题查看数据仓库的内容。

（2）元数据提供已有的可重复利用的查询语言信息。如果某个查询能够满足用户的需求，或者与用户的愿望相似，用户就可以再次使用这些查询而不必从头开始编程。

更高级的形式是用户通过选择要提出的业务问题类型来访问现有的查询，得到相似查询的元数据。

关于数据仓库使用的元数据能帮助用户到数据仓库查询所需要的信息，用于解决企业问题。

习　　题

1．画出数据仓库结构图，说明各部分内容。

2．说明数据仓库结构图中包含轻度综合数据层与高度综合数据层的作用。这些数据是根据需要临时计算的吗？

3．说明数据集市与数据仓库的区别和联系。

4．说明数据集市的特点。

5．画出数据集市的两种结构图，说明它们的不同点。

6．画出数据仓库系统结构图，说明把仓库管理和分析工具作为数据仓库系统的两个独立组成部分的原因。

7．说明仓库管理包含的具体内容。

8．说明分析工具包含的具体内容。

9．画出数据仓库的运行结构图，说明三层 C/S 结构比两层 C/S 结构的不同点。

10. 数据仓库的逻辑数据模型有哪些?
11. 说明星型模型有什么好处。
12. 说明数据仓库的数据模型为什么含时间维数据。
13. 说明雪花模型与星网模型的不同点。
14. 第三范式数据模型与星型模型有什么不同?
15. 比较第三范式与星型模型的优缺点。
16. 简单说明 ETL 过程的主要步骤。
17. 说明数据抽取工作的内容。
18. 说明数据转换的基本功能。
19. 数据转换有哪些类型?
20. 数据装载方式与类型有哪些?
21. 说明数据库中的元数据以及数据仓库中的元数据的不同。
22. 什么是关于数据源的元数据?
23. 什么是关于数据模型的元数据?
24. 什么是关于数据仓库映射的元数据?
25. 什么是关于数据仓库使用的元数据?
26. 数据仓库中的元数据是如何发挥作用的?

第3章　联机分析处理

在数据仓库系统中，联机分析处理（OLAP）是重要的数据分析工具。OLAP的基本思想是企业的决策者应能灵活地、从多方面和多角度以多维的形式来观察企业的状态和了解企业的变化。

3.1　OLAP概念

在信息爆炸的时代，信息过量几乎成为人人需要面对的问题。如何才能不被信息的汪洋大海所淹没，从中及时发现有用的知识或者规律，提高信息利用率呢？要想使数据真正成为一个决策资源，只有充分利用它为一个组织的业务决策和战略发展服务才行，否则大量的数据可能成为包袱，甚至成为垃圾。OLAP是解决这类问题的最有力的工具之一。

OLAP专门设计用于支持复杂的分析操作，侧重对分析人员和高层管理人员的决策支持，可以应分析人员的要求快速、灵活地进行大数据量的复杂查询处理，并且以一种直观易懂的形式将查询结果提供给决策制定者，以便他们准确掌握企业（公司）的经营状况，了解市场需求，制定正确方案，增加效益。OLAP软件，以它先进的分析功能和以多维形式提供数据的能力，正作为一种支持企业关键商业决策的解决方案而迅速崛起。

3.1.1　OLAP的定义

在决策活动中，决策人员需要的数据往往不是单一指标的单一的值，而是希望能够从多个角度观察某个指标或者某个值，或者找出这些指标之间的关系。比如，决策者可能想知道“东北地区和西南地区今年一季度和去年一季度在销售总额上的对比情况，并且销售额按10～50万元、50～100万元，以及100万元以上分组”。上面的问题是比较有代表性的，决策所需数据总是与一些统计指标，如销售总额、观察角度（如销售区域、时间）和不同级别的统计有关，将这些观察数据的角度称为维。可以说决策数据是多维数据，多维数据分析是决策分析的主要内容。但传统的关系数据库系统及其查询工具对于管理和应用这样复杂的数据显得力不从心。

OLAP是在OLTP的基础上发展起来的，OLTP是以数据库为基础的，面对的是操作人员和低层管理人员，对基本数据的查询和增、删、改等进行处理。而OLAP是以数据仓库为基础的数据分析处理。它有两个特点：一是在线性（on line），体现为对用户请求的快速响应和交互式操作，它的实现是由客户机/服务器这种体系结构在网络环境上完成的；二是多维分析（multi-dimension analysis），这也是OLAP的核心所在。

OLAP超越了一般查询和报表的功能，是建立在一般事务操作之上的另外一种逻辑步骤，因此，它的决策支持能力更强。在多维数据环境中，OLAP为终端用户提供了复杂的数据分析功能。高层管理人员通过OLAP能够通过浏览、分析数据去发现数据的变化趋势、

特征以及一些潜在的信息，从而更好地帮助他们了解商业活动的变化。目前，比较普遍接受的 OLAP 的定义有两种。

1. OLAP 理事会给出的定义

联机分析处理是一种软件技术，使分析人员能够迅速、一致、交互地从各个方面观察信息，以达到深入理解数据的目的。这些信息是从原始数据转换过来的，按照用户的理解，它反映了企业真实的方方面面。

企业的用户对企业的观察自然是多维的。如销售，不仅可从生产方面看，还与地点、时间等有关，这就是为什么要求 OLAP 模型是多维的原因。这种多维用户视图通过一种更为直观的分析模型进行设计和分析。

OLAP 的大部分策略都是将关系型的或普通的数据进行多维数据存储，以便于进行分析，从而达到联机分析处理的目的。这种多维数据库，也被看作超立方体沿着多个维方向存储数据，为用户沿事物的任意的多个维方向方便地分析数据。

2. OLAP 简单定义

近来，随着人们对 OLAP 理解的不断深入，有些学者提出了更为简要的定义，即联机分析处理是共享多维信息的快速分析(fast analysis of shared multidimensional information)。它体现了 4 个特征：

(1) 快速性(fast)：用户对 OLAP 的快速反应能力有很高的要求。系统应能在 5 秒内对用户的大部分分析要求作出反应，如果终端用户在 30 秒内没有得到系统的响应，则会变得不耐烦，失去分析主线索，影响分析的质量。

(2) 可分析性(analysis)：OLAP 系统应能处理与应用有关的任何逻辑分析和统计分析。尽管系统需要一些事先的编程，但并不意味着系统事先已对所有的应用都定义好了。

(3) 多维性(multidimensional)：多维性是 OLAP 的关键属性。系统必须提供对数据分析的多维视图和分析，包括对层次维和多重层次维的完全支持。

(4) 信息性(information)：不论数据量有多大，也不管数据存储在何处，OLAP 系统应能及时获得信息，并且管理大容量的信息。

用于实现 OLAP 的技术主要包括网络环境上客户机/服务器体系结构、时间序列分析、面向对象、并行处理、数据存储优化以及多线索技术等。

3.1.2 OLAP 准则

1985 年以来，关系数据库需求始终受到 E．F．Codd 提出的十二条准则的影响。1993 年，E. F. Codd 在《Providing OLAP to User Analysts》中又提出了有关 OLAP 的十二条准则，用来评价分析处理工具，这也是他继关系数据库和分布式数据库提出的两个“十二条准则”后提出的第三个“十二条准则”。由于这些准则最初是对客户研究的结果，所以业界对这个十二条准则褒贬不一。但其主要方面，如多维数据分析、客户/服务器结构、多用户支持及一致的报表性能等方面还是得到了大多数人的认可。E. F. Codd 在文中系统阐述了有关 OLAP 产品及其所依赖的数据分析模型的一系列概念及衡量标准，这对 OLAP 产品的辨别

及后来发展方向的确立都产生了重要的作用。如今，这十二条准则也成为大家定义 OLAP 的主要依据，被认为是 OLAP 产品应该具备的特征。如今 OLAP 的概念已经在商业数据库领域得以广泛使用，Codd 提出的 OLAP 准则如下。

1. 多维概念视图

从用户分析员的角度来看，用户通常按多维角度来看待企业，企业决策分析的目的不同，决定了分析和衡量企业的数据总是从不同的角度来进行，所以企业数据空间本身就是多维的。因此 OLAP 的概念模型也应是多维的。用户可以简单、直接地操作这些多维数据模型。例如，用户可以对多维数据模型进行切片、切块、改变坐标或旋转模式中的联合（概括和聚集）数据路径。

2. 透明性

透明性原则包括两层含义：首先，OLAP 在体系结构中的位置对用户是透明的。OLAP 应处于一个真正的开放系统结构中，可使分析工具嵌入用户所需的任何位置，而不会对分析工具的使用产生副作用。同时必须保证 OLAP 工具的嵌入不会引入和增加任何复杂性。其次，OLAP 的数据源对用户也是透明的。用户只需使用熟悉的查询工具进行查询，而不必关心 OLAP 工具获取的数据是来自于同质还是异质的数据源。

3. 可访问性

OLAP 系统不仅能进行开放的存取，而且还能提供高效的存取策略。OLAP 用户分析员不仅能在公共概念视图的基础上对关系数据库中的数据进行分析，而且在公共分析模型的基础上还可以对关系数据库、数据仓库的数据进行分析。要实现这些功能，就要求 OLAP 能将自己的概念视图映射到异质的数据存储上，并可访问数据，还能进行所需的转换以便给出单一的、连贯的、一致的用户视图。另外必须说明的一点就是，物理数据来源于何种系统，这对用户来说应是透明的，进行处理的是 OLAP 工具而不是用户分析员。这是提供 OLAP 工具透明性准则的基础之一。

OLAP 系统应该提供高效的存储策略，使系统只存取与指定分析有关的数据，避免多余的数据存取。

4. 一致稳定的报表性能

报表操作不应随维数增加而削弱，即当数据维数和数据的综合层次增加时，提供给最终分析员的报表能力和响应速度不应该有明显的降低，这对维护 OLAP 产品的简易性至关重要。即便是企业模型改变时，关键数据的计算方法也无需更改。也就是说，OLAP 系统的数据模型对企业模型应该具有“鲁棒”性。只有做到这一点，OLAP 工具提供的数据报表和所做的预测分析的结果才是可信的。

5. 客户/服务器体系结构

OLAP 是建立在客户/ 服务器体系结构上的，要求它的多维数据库服务器能够被不同

的应用和工具所访问,服务器端智能地以最小的代价完成同多种服务器之间的挂接任务,智能化服务器必须具有在不同的逻辑的和物理的数据库间映射并组合数据的能力,还应构造通用的、概念的、逻辑的和物理的模式。从而保证透明性和建立统一的公共概念模式、逻辑模式和物理模式。客户端负责应用逻辑及用户界面。

6. 维的等同性

每一数据维在其结构和操作功能上必须等价。可能存在适用于所有维的逻辑结构,提供给某一维的任何功能也应提供给其他维。即系统可以将附加的操作能力授给所选维,但必须保证该操作能力可以授给任意的其他维,即要求维上的操作是公共的。该准则实际上是对维的基本结构和维上的操作要求。

7. 动态的稀疏矩阵处理

OLAP 服务器的物理结构应完全适用于特定的分析模式,创建和加载此种模式是为了提供优化的稀疏矩阵处理。当存在稀疏矩阵时,OLAP 服务器应能推知数据是如何分布的,以及怎样存储才更有效。

该准则包括两层含义:第一,对任意给定的稀疏矩阵,存在一个最优的物理视图,该视图能提供最大的内存效率和矩阵处理能力,稀疏度是数据分布的一个特征,不能适应数据集合的数据分布将会导致快速、高效操作的失败。第二,OLAP 工具的基本物理数据单元可配置给可能出现的维的子集。同时,还要提供动态可变的访问方法并包含多种存取机制,例如,直接计算地址、B 树索引、导出算法、哈希算法或这些技术的最佳组合。访问速度不会因数据维的多少、数据集的大小而变化。

如果分析要求较为单一和固定,那么确实有可能针对它建立起一个最优的、静态的、具有固定维数的物理模式。但实际上,分析需求的特点就是具有不确定性,所以建立静态模式是不现实的,因此 OLAP 工具必须使得模型的物理模式充分适应指定的维数,尤其是特定模型的数据分布。

8. 多用户支持能力

当多个用户在同一分析模式上并行工作,或是在同一企业数据上建立不同的分析模型时,OLAP 工具应提供并发访问、数据完整性及安全性等功能。

实际上,OLAP 工具必须支持多用户也是为了适合数据分析工作的特点。应该鼓励以工作组的形式来使用 OLAP 工具,这样多个用户可以交换各自的想法和分析结果。

9. 非限定的跨维操作

在多维数据分析中,所有维的生成和处理都是平等的。OLAP 工具应能处理维间相关计算。如果计算时需要按语言定义各种规则,此种语言应允许计算和数据操作跨越任意数目的数据维,而不必限制数据单元间的任何关系,也不必考虑每一单元包含的通用数据属性数目。

10. 直观的数据操作

OLAP 操作要求直观易懂。如果要重定向联系路径，或在维或行间进行细剖操作，都应该通过直观的操作分析模型来完成，而不需要使用菜单，也不需要跨越用户界面进行多次操作，即综合路径重定位、向上综合、向下钻取和其他操作都可以通过直观、方便的点、拉操作完成。

在分析模型中定义的维应包含用户分析所需的所有信息，从而可以进行任意继承操作。

11. 灵活的报表生成

用户使用 OLAP 服务器及其工具，可以按任何想要的方式来操作、分析、综合和查看数据，这些方式包括将行、列及单元按需要依次排放。报表机制也应提供此种灵活性，报表必须能从各种可能的方面显示出从数据模型中综合出的数据和信息，充分反映数据分析模型的多维特征，并可按用户需要的方式来显示它。

12. 不受限制的维和聚集层次

OLAP 服务器应能在通用分析模型中协调至少 15 个维。每一通用维应能允许有任意个用户定义的聚集，而且用户分析员可以在任意给定的综合路径上建立任意多个聚集层次。

3.1.3 OLAP 的基本概念

OLAP 是针对特定问题的联机数据访问和分析的。通过对信息进行快速、稳定一致和交互性的存取，允许管理决策人员对数据进行深入观察。为了对 OLAP 技术有更深入的了解，这里主要介绍在 OLAP 中常用的一些基本概念。

(1) 变量

变量是数据的实际意义，即描述数据“是什么”。例如，数据“100”本身并没有意义或者说意义未定，可能是一个学校的学生人数，也可能是某产品的单价，还可能是某商品的销售量，等等。一般情况下，变量是一个数值度量指标，例如，“人数”、“单价”、“销售量”等都是变量，而“100”则是变量的一个值。

(2) 维

维是人们观察数据的特定角度。例如，企业常常关心产品销售数据随着时间推移而产生的变化情况，这时是从时间的角度来观察产品的销售，所以时间是一个维(时间维)。企业也时常关心自己的产品在不同地区的销售分布情况，这时是从地理分布的角度来观察产品的销售，所以地理分布也是一个维(地理维)。其他还有如产品维、顾客维等。

(3) 维的层次

人们观察数据的某个特定角度(即某个维)还可以存在细节程度不同的多个描述方面，称这多个描述方面为维的层次。一个维往往具有多个层次，例如，描述时间维时，可以从日期、月份、季度、年等不同层次来描述，那么日期、月份、季度、年等就是时间维的层次。同样，

城市、地区、国家等构成了地理维的层次。

(4) 维成员

维的一个取值称为该维的一个维成员。如果一个维是多层次的，那么该维的维成员由各个不同维层次的取值组合而成。例如，考虑时间维具有日期、月份、年这3个层次，分别在日期、月份、年上各取一个值组合起来，就得到了时间维的一个维成员，即“某年某月某日”。一个维成员并不一定在每个维层次上都要取值，例如，“某年某月”、“某月某日”、“某年”等都是时间维的维成员。对应一个数据项来说，维成员是该数据项在某维中位置的描述。例如对一个销售数据来说，时间维的维成员“某年某月某日”就表示该销售数据是“某年某月某日”的销售数据，“某年某月某日”是该销售数据在时间维上位置的描述。

(5) 多维数组

一个多维数组可以表示为：(维1，维2，…，维n，变量)。例如，若日用品销售数据是按时间、地区和销售渠道组织起来的三维立方体，加上变量“销售额”，就组成了一个多维数组(地区，时间，销售渠道，销售额)，如果在此基础上再扩展一个产品维，就得到一个四维的结构，其多维数组为(产品，地区，时间，销售渠道，销售额)。

(6) 数据单元(单元格)

多维数组的取值称为数据单元。当多维数组的各个维都选中一个维成员，这些维成员的组合就惟一确定了一个变量的值。那么数据单元就可以表示为：(维1维成员，维2维成员，…，维n维成员，变量的值)。例如，在产品、地区、时间和销售渠道上各取维成员“牙膏”、“上海”、“2004年12月”和“批发”，就惟一确定了变量“销售额”的一个值(假设为100 000)，则该数据单元可表示为：(牙膏，上海，2004年12月，批发，100 000)。

3.2 OLAP的数据模型

建立OLAP的基础是多维数据模型，多维数据模型的存储可以有多种不同的形式。MOLAP和ROLAP是OLAP的两种主要形式，其中MOLAP(multi-dimension OLAP)是基于多维数据库的OLAP，简称为多维OLAP；ROLAP(relation OLAP)是基于关系数据库的OLAP，简称关系OLAP。还有几种OLAP，如WOLAP(Web OLAP)代表网络OLAP，HOLAP(hybrid OLAP)代表混合OLAP。

3.2.1 MOLAP数据模型

MOLAP数据模型是基于多维数据库的OLAP，多维数据库(multi dimensional database，MDDB)是以多维方式组织数据，即以维作为坐标系，采用类似于数组形式存储数据。多维数据库中的元素具有相同类型的数值，如销售量。例如，二维MDDB(数组)的数据组织见表3.1所示。它代表不同产品(衣服、鞋、帽)在不同地区(北京、上海、广州)的销售量情况。

表 3.1　MDDB(二维)数据组织

项目＼地区	北京	上海	广州
衣服	600	700	500
鞋	800	900	700
帽子	100	200	80

在查询中除查询一般的"衣服在广州的销售量"外,有时查询像"衣服的总销售量"等类问题,涉及多个数据项求和,如果采取临时进行累加计算,会使查询效率大大降低,为此,需要增加汇总数据项。在多维数据库中只需要按行或列进行求和,增加"总和"的维成员即可,见表 3.2 所示。

表 3.2　多维数据库中含综合数据的数据组织

项目＼地区	北京	上海	广州	总和
衣服	600	700	500	1800
鞋	800	900	700	2400
帽子	100	200	80	380
总和	1500	1800	1280	4580

MDDB 的数据组织形式不同于关系数据库的组织形式,关系数据库是以"属性-元组(记录)"形式组织数据。对表 3.1 中的数据按关系数据库组织,数据见表 3.3 所示。

表 3.3　关系数据库 RDBMS 数据组织

产品名	地区	销售量
衣服	北京	600
衣服	上海	700
衣服	广州	500
鞋	北京	800
鞋	上海	900
鞋	广州	700
帽子	北京	100
帽子	上海	200
帽子	广州	80

可见,多维数据库 MDDB 比关系数据库表达更清晰且占用的存储少。在关系数据库中增加综合数据项,见表 3.4 所示。这些综合数据项一般在建立数据库时,同时计算出来。这样在查询时,不必临时进行计算,提高了查询效率。对于多维数据库的综合数据项明显比关系数据库的综合项更有效果。

表 3.4 关系数据库中综合数据的数据组织

产品名	地区	销售量
衣服	北京	600
衣服	上海	700
衣服	广州	500
衣服	总和	1800
鞋	北京	800
鞋	上海	900
鞋	广州	700
鞋	总和	2400
帽子	北京	100
帽子	上海	200
帽子	广州	80
帽子	总和	380

3.2.2 ROLAP 数据模型

ROLAP 是基于关系数据库的 OLAP,见表 3.3 所示。它是一个平面结构,用关系数据库表示多维数据时,采用星型模型,即用两类表,一类是事实表,存储事实的实际值,如销售量;另一类是维表,对每一个维来说,至少有一个表来存储该维的描述信息,如产品的名称、分类等。星型模型完全用二维关系表示了数据的多维观念。

通过关系数据库实现多维查询时,通过维表的主码对事实表和每一个维表做连接操作,一次查询就可以得到数据的具体值以及对数据的多维描述(即对应的各维上的维成员)。但是,因为对每个维都需要进行一次连接操作,所以系统的性能就成了 ROLAP 实现的最大的一个问题,特别是当维数增加和事实表增大时,必须采用有效的查询优化技术(特别是表连接策略),利用各种索引技术来提高系统的性能。

对于存在多层次的复杂维时,需要采用"雪花模型",用多张表来描述一个复杂维。对于存在综合数据时,需要建立汇总事实表,采用"星网模型"来描述。

3.2.3 MOLAP 与 ROLAP 的比较

MOLAP 通过多维数据库引擎从关系数据库(DB)和数据仓库(DW)中提取数据,将各种数据组织成多维数据库,存放到 MDDB 中,并将自动建立索引,并进行预综合,来提高查询存取性能,如图 3.1 所示。

ROLAP 从关系数据库(DB)和数据仓库(DW)中提取数据,按关系 OLAP(ROLAP)的数据组织存放在关系数据库服务器(RDBMS 服务器)中。最终用户的多维分析请求,通过 ROLAP 服务器的多维分析引擎动态翻译成 SQL 请求,将查询结果经多维处理(将关系表达式转换成多维视图)返回用户,如图 3.2 所示。

虽然这两种技术都满足了 OLAP 数据处理的一般过程:即数据装入、汇总、建索引和提供使用,但 MOLAP 较 ROLAP 要简明一些,MOLAP 的索引及数据综合可以自动进行。然而 ROLAP 的实现较为复杂,但灵活性较好,用户可以动态实现统计或计算方式。

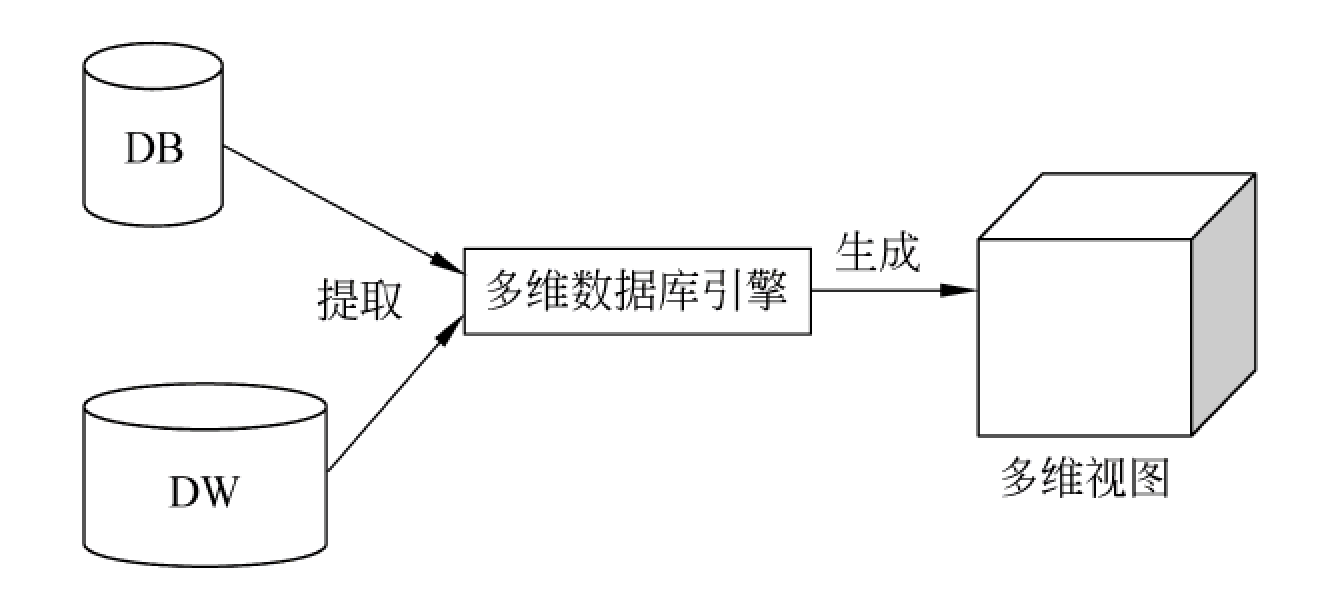


图 3.1　MOLAP 结构

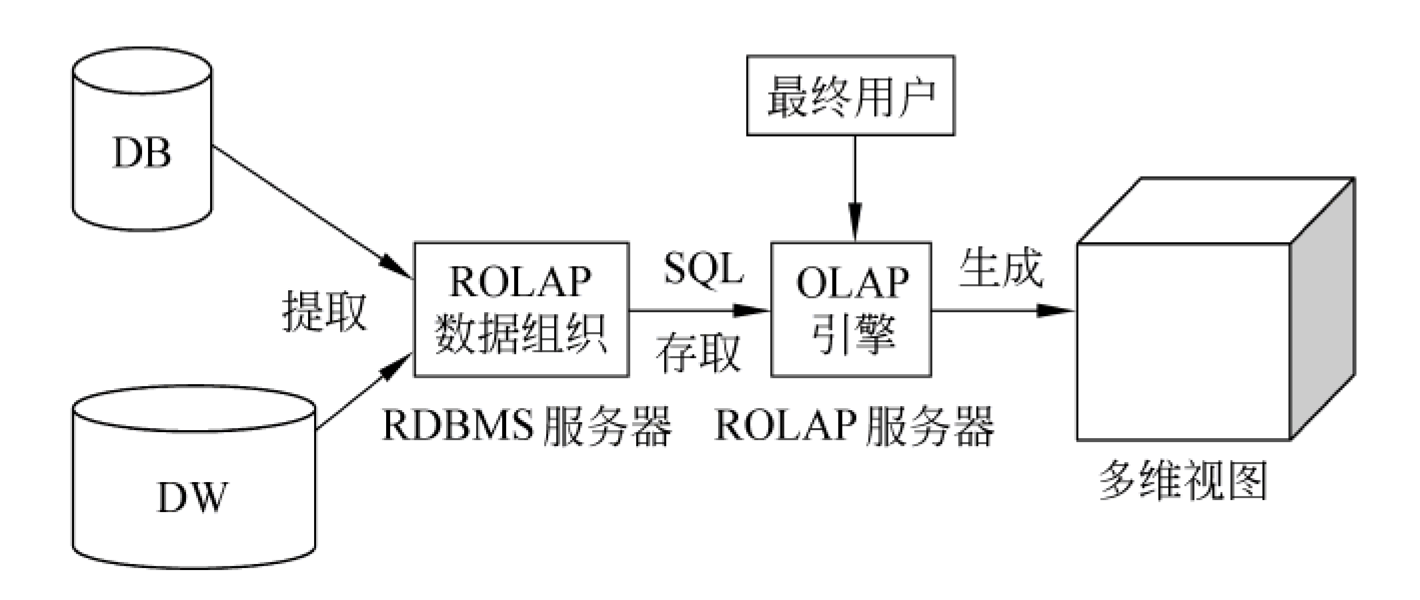


图 3.2　ROLAP 结构

下面详细深入分析 MOLAP 与 ROLAP 的对比。

1. 数据存取速度

ROLAP 的多维数据是以星型模型等关系数据库(平面形式)存储,并不直接体现"超立方体"形式。在接收客户 OLAP 请求时,ROLAP 服务器需要将 SQL 语句转化为多维存储语句,并利用连接运算临时"拼合"出多维数据立方体,因此,ROLAP 的响应时间较长。

目前,关系型数据库已经对 OLAP 做了很多优化,包括并行存储、并行查询、并行数据管理、基于成本的查询优化、位图索引、SQL 的 OLAP 扩展等,大大提高了 ROLAP 的速度。

MOLAP 是专为 OLAP 所设计,能够自动地建立索引,并且有良好的预计算能力,能够使用多维查询语句访问数据立方体,因此 MOLAP 在数据存储速度上性能好,响应速度快。

2. 数据存储的容量

ROLAP 使用的传统关系数据库的存储方法,存储容量基本上没有限制。但是,需要指出的是,在 ROLAP 中为了提高分析响应速度,常常构造大量的中间表(如综合表),这些中间表带来了大量的冗余数据。

MOLAP 通常采用多平面叠加成立体的方式存放数据,(这样访问速度快),由于受操作系统平台中文件大小的限制,当数据量超过操作系统最大文件长度时,需要进行数据分割。随着数量的增大,多维数据库进行的预运算结果将占用巨量的空间,此时可能导致"数据爆炸"的现象。因此,多维数据库的数据量级难以达到太大的字节级。

3. 多维计算的能力

MOLAP 能够支持高性能的决策支持计算，包括复杂的跨维计算、行级的计算，而在 ROLAP 中，SQL 无法完成部分计算，并且 ROLAP 无法完成多行的计算和维之间的计算。

4. 维度变化的适应性

MOLAP 需要在建立多维数据库前确定各个维度以及维度的层次关系。在多维数据库建立之后，如果要增加新的维度，则多维数据库通常需要重新建立。新增维度数据会急剧增加。而 ROLAP 增加一个维度，只是增加一张维表并修改事实表，系统中其他维表不需要修改，因此 ROLAP 对于维表的变更有很好的适应性。

5. 数据变化的适应性

由于 MOLAP 通过预综合处理来提高速度，当数据频繁地变化时，MOLAP 需要进行大量的重新计算，甚至重新建立索引乃至重构多维数据库。在 ROLAP 中，预综合处理通常由设计者根据需求制定，因此灵活性较好，对于数据变化的适应性高。

6. 软硬件平台的适应性

由于关系数据库已经在众多的软硬件平台上成功地运行，即 ROLAP 对软硬件平台的适应性很好，而 MOLAP 相对较差。

7. 元数据管理

元数据是 OLAP 和数据仓库的核心数据，OLAP 的元数据包括层次关系、计算转化信息、报表中的数据项描述、安全存取控制、数据更新、数据源和预计算综合表等，目前在元数据的管理上，MOLAP 和 ROLAP 都没有成形的标准，MOLAP 产品将元数据作为其内在数据，而 ROLAP 产品将元数据作为应用开发的一部分，由设计者来定义和处理。

MOLAP 和 ROLAP 在技术上各有优缺点。MOLAP 以多维数据库为核心，在数据存储和综合上有明显的优势，但它不适应太大的数据存储，特别对有大量稀疏数据的存储将会浪费大量的存储空间。ROLAP 以 RDBMS 为基础，利用成熟的技术为用户的使用和管理带来方便。

MOLAP 和 ROLAP 在数据存储、技术和特性的比较，见表 3.5 所示。

表 3.5　MOLAP 和 ROLAP 的比较

项目	数据存储	技　术	特　征
MOLAP	详细数据用关系表存储在数据仓库中；各种汇总数据保存在多维数据库中；从数据仓库中询问详细数据，从多维数据库中询问汇总数据	由 MOLAP 引擎创建；预先建立数据立方体；多维视图存储在陈列中，而不是表格中；可以高速检索矩阵数据；利用稀疏矩阵技术来管理汇总的稀疏数据	询问响应速度快；能轻松适应多维分析；有广泛的下钻和多层次/多视角的查询能力

续表

项目	数据存储	技　术	特　征
ROLAP	全部数据以关系表存储在数据仓库中；可获得细节的和综合汇总的数据；有非常大的数据容量；从数据仓库中询问所有的数据	使用复杂 SQL 从数据仓库中获取数据；ROLAP 引擎在分析中创建数据立方体；表示层能够表示多维的视图	在复杂分析功能上有局限性，需要采用优化的 OLAP；向下钻取较容易，但是跨维向下钻取比较困难

3.2.4　HOLAP 数据模型

HOLAP(hybrid OLAP)，即混合 OLAP 介于 MOLAP 和 ROLAP 之间。在 HOLAP 中，对最常用的维度和维层次使用多维数据库来存储，对于用户不常用的维度和数据，采用 ROLAP 星型结构来存储。当用户询问不常用数据时，HOLAP 将会把简化的多维数据库和星型结构进行拼合，从而得到完整的多维数据库。

在 HOLAP 的多维数据库中的数据维度少于 MOLAP 中的维度，数据存储容量也少于 MOLAP 方式。但是，HOLAP 在数据存取速度上又低于 MOLAP。

3.3　多维数据的显示

3.3.1　多维数据的显示方法

多维数据一般采用多维数据库(MDDB)和关系数据库(RDBMS)两种方式存储数据。多维数据的显示只能在平面上展现出来。对于二级数据采用多数据库形式显示时，见表3.1 所示。若增加一维时间维，就无法在平面上展现出来。二维数据采用关系数据库形式显示时，见表 3.3 所示。若增加一维时间维，仍然可以显示出来，见表 3.6 所示。

表 3.6　三维数据的关系数据库显示

产品名	地区	时间	销售量
衣服	北京	1 月	100
衣服	北京	2 月	200
衣服	北京	3 月	300
衣服	上海	1 月	200
衣服	上海	2 月	300
衣服	上海	3 月	400
衣服	广州	1 月	150
衣服	广州	2 月	250
衣服	广州	3 月	300
鞋	北京	1 月	150
鞋	北京	2 月	300

续表

产品名	地区	时间	销售量
鞋	北京	3 月	350
鞋	上海	1 月	200
鞋	上海	2 月	300
鞋	上海	3 月	400
鞋	广州	1 月	150
鞋	广州	2 月	250
鞋	广州	3 月	300
…	…	…	…

用关系数据库可以显示更多维的数据，即用星型模型的事实表形式显示。但是，用事实表显示多维数据时，重复数据很多，也显得很烦琐。

用多维数据库显示时，虽然不能同时显示三维以上数据，由于显示的数据很精练，所以仍然用多维数据库的方式来显示多维数据。一般在多维数据库中，固定一些维成员，重点显示两维的数据。如在表 3.6 三维数据中，固定地区维是“北京”时的两维数据的显示，如表 3.7 所示。

表 3.7　北京地区销售情况表

北京地区	1 月	2 月	3 月
衣服	100	200	300
鞋子	150	300	350
…	…	…	…

3.3.2　多维类型结构

为了有效地表示多维数据，E. Thomsen 引入多维类型结构（MTS），有些专家称为多维域结构（MDS）。表示方法是：每一个维度用一条线段来表示。维度中的每一个成员都用线段上的一个单位区间来表示。例如，用 3 个线段分别表示时间、产品和指标 3 个维的多维类型结构，如图 3.3 所示。

在图 3.3 多维类型结构中，指定时间维成员是 3 月，产品维成员是鞋，指标维成员是销售量，这样代表了三维数据总的一个空间数据点，如图 3.4 所示。

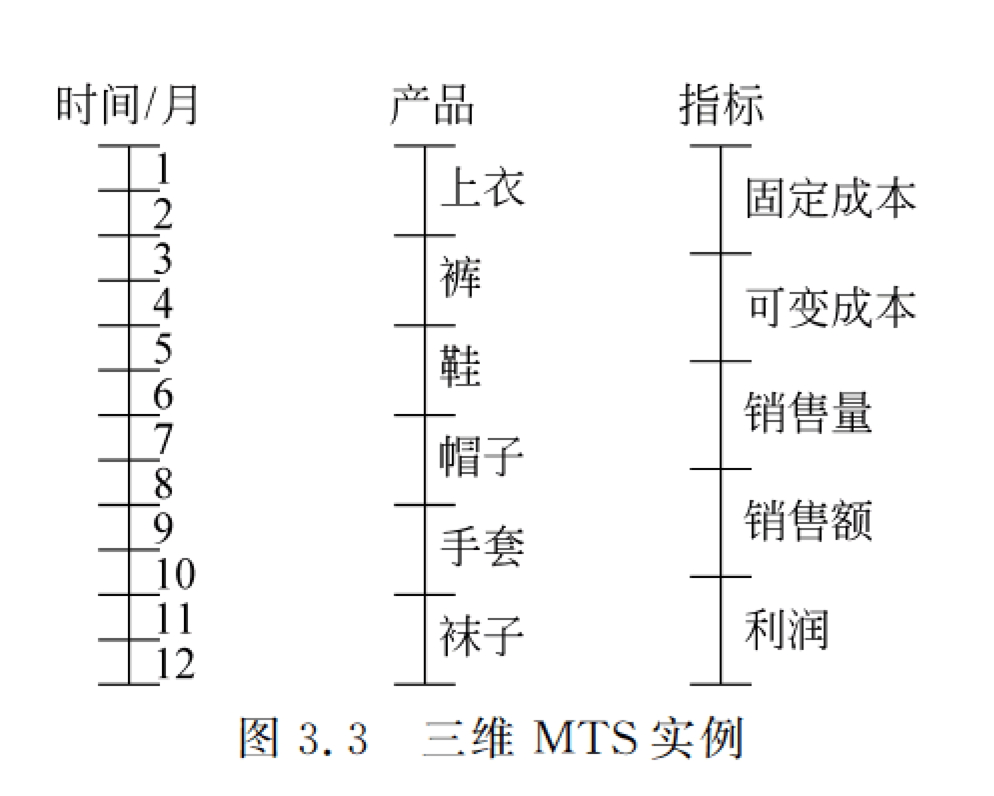


图 3.3　三维 MTS 实例

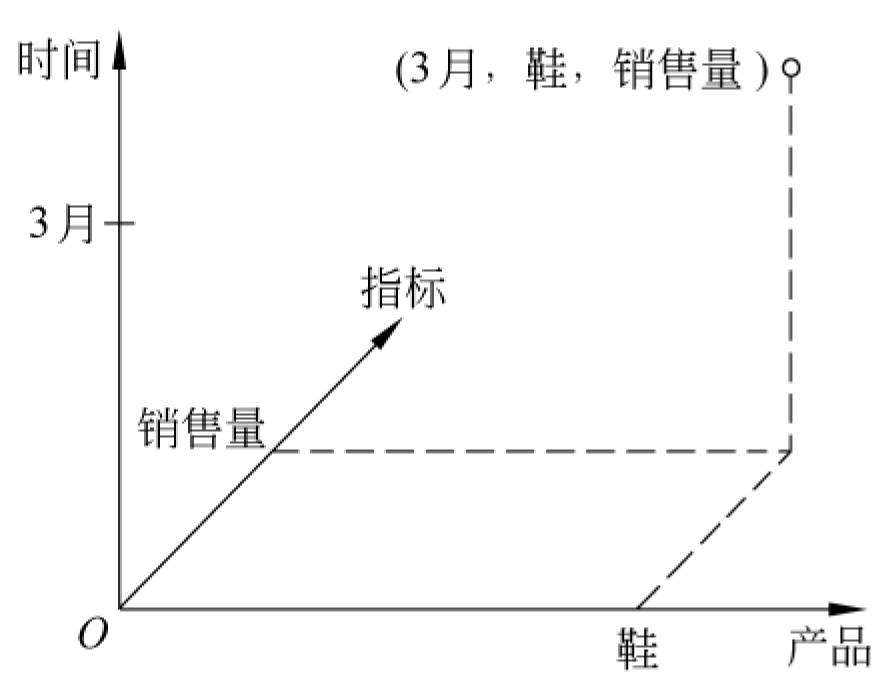


图 3.4　多维类型结构中的空间数据点

在 MTS 中,在原有多维数据中增加一个维是很容易的,例如在图 3.3 的三维中增加一个商店维,这时需要增加一个线段表示商店维,如图 3.5 所示。

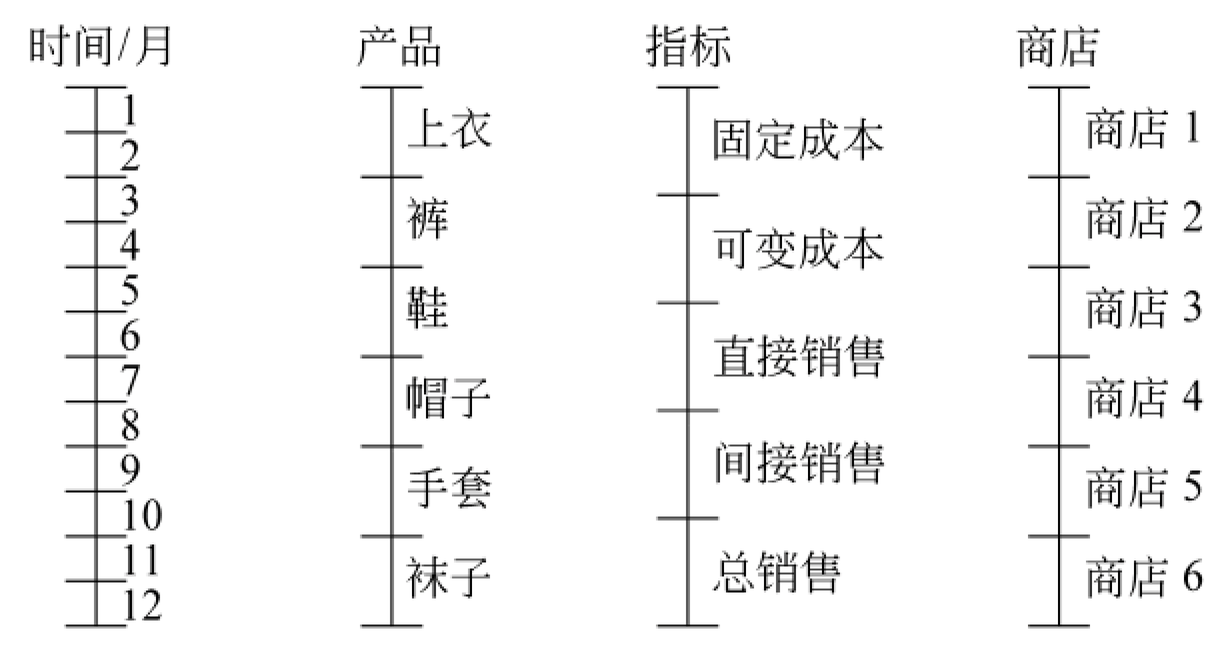


图 3.5　四维 MTS 实例

3.3.3　多维数据的分析视图

在平面的屏幕上显示多维数据,是利用行、列和页面 3 个显示组来表示的。例如,对图 3.5 中的四维 MTS 实例,在页面上选定商店维度中“商店 3”,在行中选定时间维的“1 月、2 月、3 月”共 3 个成员,在列中选定产品维中的“上衣、裤、帽子”3 个成员,以及指标维中的“固定成本、直接销售”2 个成员。该四维数据的显示如图 3.6 所示。

商店 3 (页面)	上衣		裤		帽子	
	直接销售	固定成本	直接销售	固定成本	直接销售	固定成本
1 月	450	350	550	450	500	400
2 月	380	280	460	360	400	320
3 月	400	310	480	410	450	400

图 3.6　四维数据的显示

对于更多维度的数据显示,需要选择维度及其成员分布在行或者列中。在页面上可以选定多个维度,但每个维度只能显示一个成员。在行或者列中一般只选择 2 个维,每个维可以有多个成员。例如,对 6 个维度数据,它的 MTS 如图 3.7 所示。

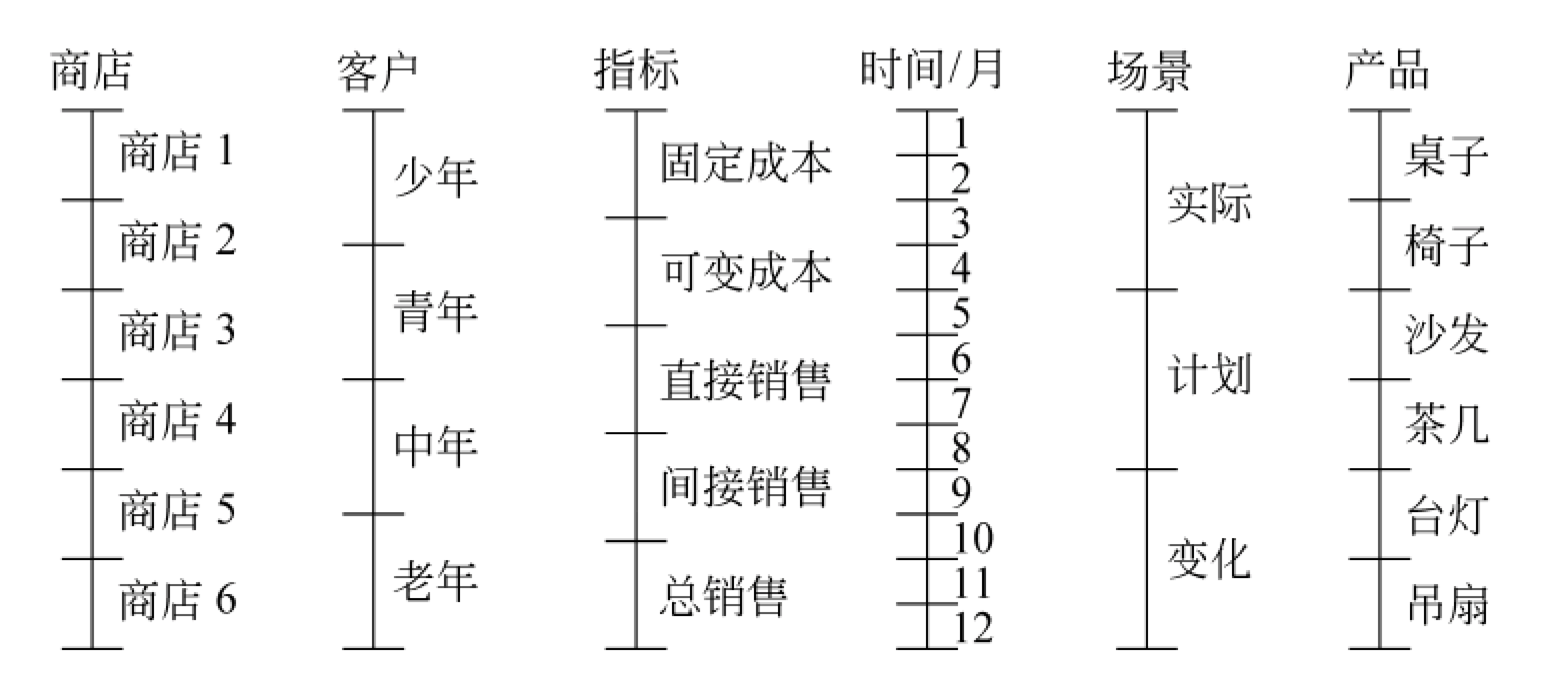


图 3.7　六维 MTS 实例

对以上六维数据中，设定页面维度为商店的成员是“商店 3”，客户维度成员是“老年”。行维度含时间维和产品维共 2 个维度，其中时间维中成员为“1 月、2 月、3 月”。产品维中成员为“桌子、台灯”。列维度含指标维和场景维共 2 个维度，其中指标维中成员为“直接销售、间接销售、总销售”。场景维中成员为“实际、计划”。具体的显示数据如图 3.8 所示。

商店 3，老年（页面）		直接销售		间接销售		总销售	
		实际	计划	实际	计划	实际	计划
1 月	桌子	250	300	125	150	375	450
	台灯	265	320	133	160	400	480
2 月	桌子	333	400	167	200	500	600
	台灯	283	340	142	170	425	510
3 月	桌子	350	420	175	210	525	630
	台灯	250	300	125	150	375	450

图 3.8　六维数据的显示

由于整个屏幕的空间是有限的，将维度嵌套在行或者列中相对于放在页维度中会占据更多的屏幕空间。用于显示维度的空间越多，则用于显示数据的空间就会越少。随着显示数据空间的减少，为了查看同样的数据就需要做更多的卷屏操作。卷屏操作的增加也加大了理解正在寻找的数据的难度。一些经验规则如下：

（1）将维度尽量放在页中，除非确定需要同时看到一个维度的多个成员。让屏幕上的信息尽量相关。

（2）当维度嵌套在行或者列中时，考虑到垂直空间比水平空间更有用，所以将维度嵌套在列中比嵌套在行中要好。一个经典的显示方法就是在行上有 1 个维度，而在列上嵌套 1 到 3 个维度，而其他的维度则放在页中，如图 3.6 所示。

（3）在决定数据的屏幕显示方式之前，应该首先弄清楚需要查找和分析比较的内容。例如，如果需要比较某个产品和某类客户在商品和时间上的实际成本情况，可以将产品和客户放在页面维度中，而在屏幕上则可以按商店和时间来显示实际成本，如图 3.9 所示。

页面维度：产品维成员“鞋”，指标维成员“成本”，场景维成员“实际”，客户维成员“青年”。

商店 \ 月份	1 月	2 月	3 月	4 月
商店 1	125	170	157	114
商店 2	200	195	129	157
商店 3	136	158	132	144

图 3.9　按照商店和时间比较成本的数据组织

3.4 OLAP 的多维数据分析

3.4.1 多维数据分析的基本操作

OLAP 的目的是为决策管理人员通过一种灵活的多维数据分析手段，提供辅助决策信息。基本的多维数据分析操作包括切片、切块、旋转、钻取等。随着 OLAP 的深入发展，OLAP 也逐渐具有了计算和智能的能力，这些能力称为广义 OLAP 操作。

1. 切片（slice）

选定多维数组的一个二维子集的操作叫做切片，即选定多维数组（维 1，维 2，…，维 n，变量）中的两个维：如维 i 和维 j，在这两个维上取某一区间或任意维成员，而将其余的维都取定一个维成员，则得到的就是多维数组在维 i 和维 j 上的一个二维子集，称这个二维子集为多维数组在维 i 和维 j 上的一个切片，表示为：（维 i，维 j，变量）。

切片就是在某两个维上取一定区间的维成员或全部维成员，而在其余的维上选定一个维成员的操作。这里可以得出两点共识：维是观察数据的角度，那么切片的作用或结果就是舍弃一些观察角度，使人们能在两个维上集中观察数据。因为人的空间想象能力毕竟有限，一般很难想象四维以上的空间结构。所以对于维数较多的多维数据空间，数据切片是十分有意义的。

图 3.10 所示是一个按产品维、地区维和时间维组织起来的产品销售数据，用三维数组表示为：（地区，时间，产品，销售额）。如果在地区维上选定一个维成员（设为“上海”），就得到了在地区维上的一个切片（关于“时间”和“产品”的切片）；在产品维上选定一个维成员（设为“电视机”），就得到了在产品维上的一个切片（关于“时间”和“地区”的切片）。显然，这样的切片数目取决于每个维上维成员的个数。

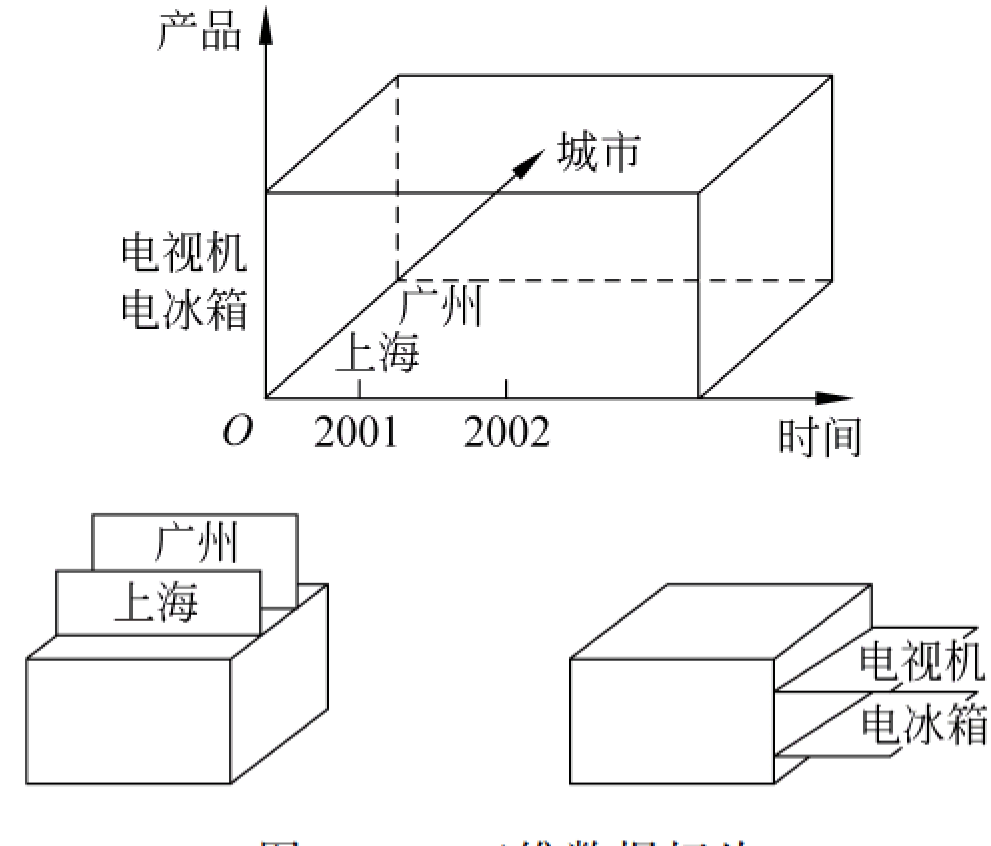


图 3.10　三维数据切片

2. 切块（dice）

切块有两种情况：

（1）在多维数组的某一个维上选定某一区间的维成员的操作

切块可以看成是在切片的基础上，确定某一个维成员的区间得到的片段，也即由多个切片叠合起来。对于时间维的切片（时间取一个确定值），如果将时间维上的取值设定为一个区间（例如取“2001—2005 年”），就得到一个数据切块，可以看成由 2001 年至 2005 年 5 个切片叠合而成的。

（2）选定多维数组的一个三维子集的操作

在多维数组（维 1，维 2，…，维 n，变量）中选定 3 个维，维 i、维 j、维 k，在这 3 个维上分别

取一个区间，或任意维成员，而其他维都取定一个维成员。如在三维数组（地区、时间、产品、销售额）中地区维取上海与广州两个维成员，产品维取电视机、电冰箱两个维成员，时间维取2003至2005的区间（3个维成员）组成三维立方体，见图3.11所示。

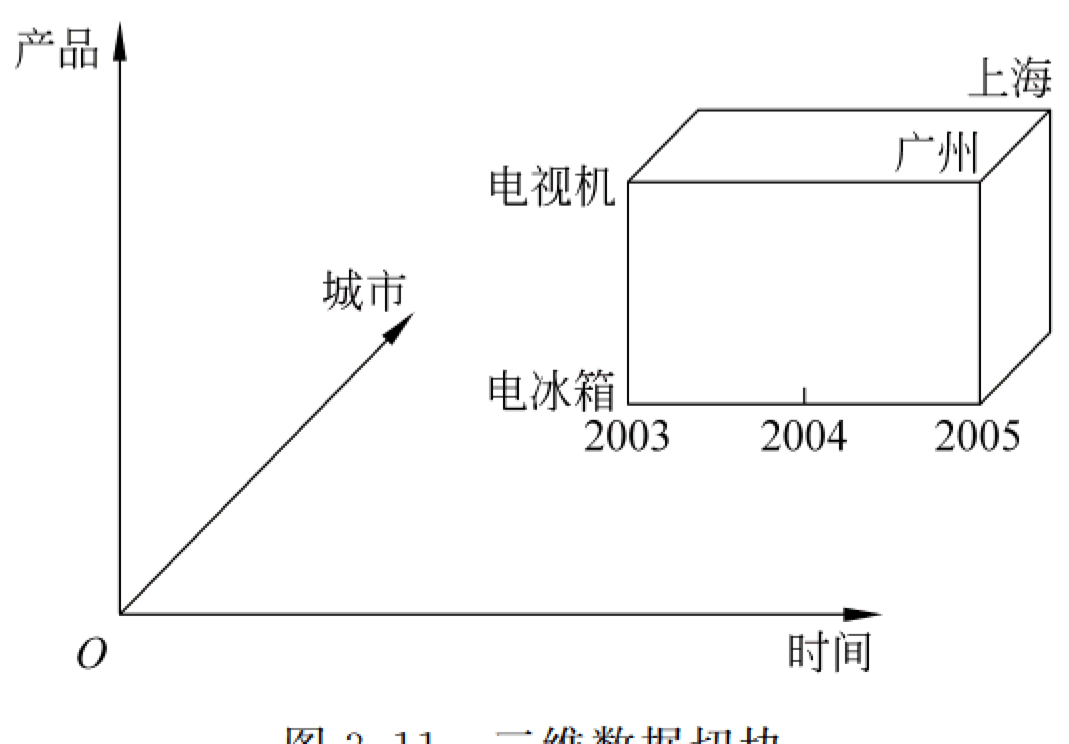


图 3.11　三维数据切块

3. 钻取（drill）

钻取有向下钻取（drill down）和向上钻取（drill up）操作。向下钻取是使用户在多层数据中能通过导航信息而获得更多的细节性数据，而向上钻取获取概括性的数据。例如2005年各部门销售收入，如表3.8所示。

表 3.8　部门销售数据

部门	销售
部门 1	900
部门 2	650
部门 3	800

在时间维进行下钻操作，获得新表3.9。

表 3.9　部门销售下钻数据

项目	2005 年			
部门	1 季度	2 季度	3 季度	4 季度
部门 1	200	200	350	150
部门 2	250	50	150	150
部门 3	200	150	180	270

相反的操作为上钻。钻取的深度与维所划分的层次相对应。

4. 旋转（pivot）

通过旋转可以得到不同视角的数据。旋转操作相当于平面数据将坐标轴旋转。例如，旋转可能包含了交换行和列，或是把某一个行维移到列维中去，或是把页面显示中的一个维和页面外的维进行交换（令其成为新的行或列中的一个），如图3.12所示。

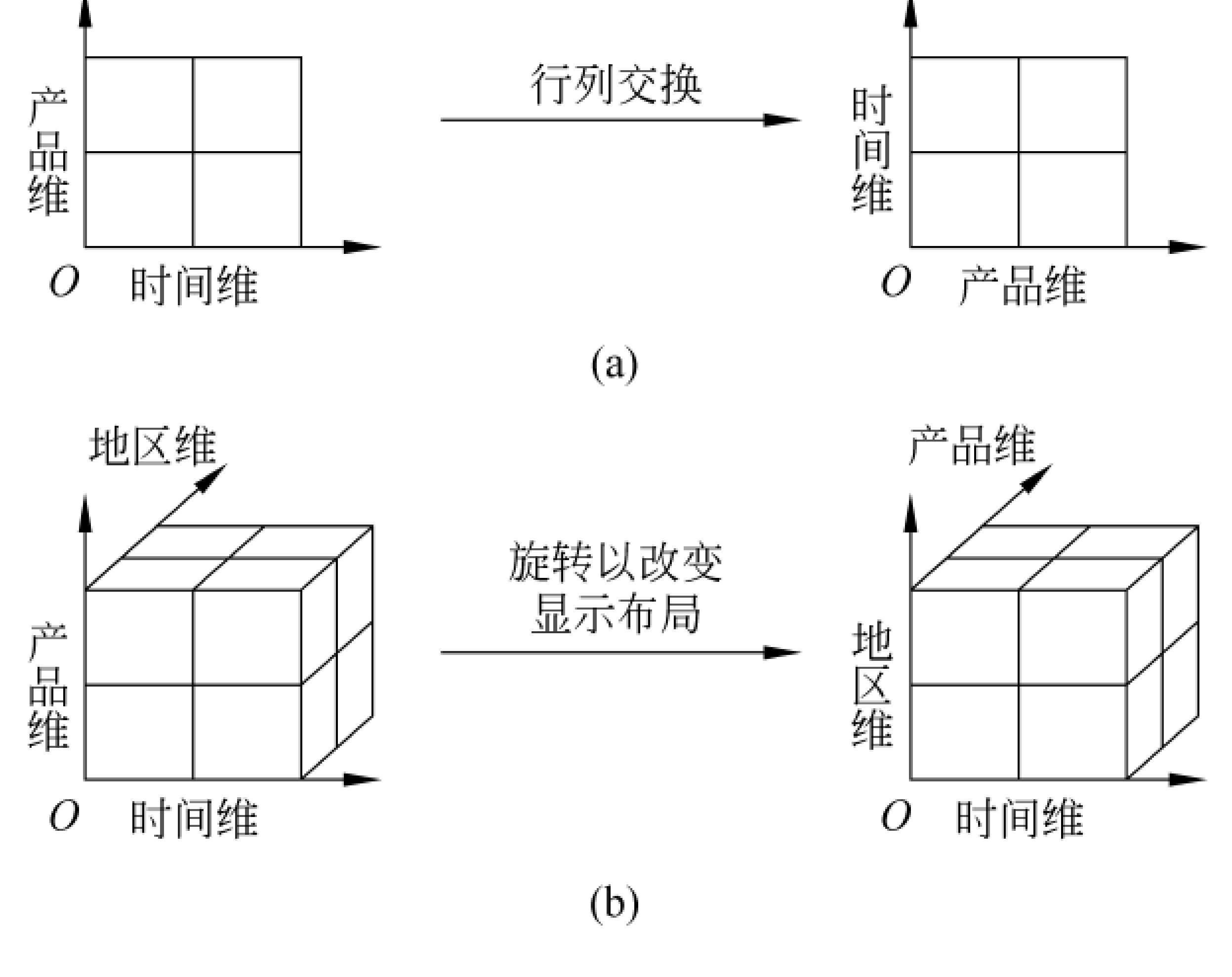


图 3.12 旋转操作

图 3.12(a)是把一个横向为时间、纵向为产品的报表旋转成为横向为产品、纵向为时间的报表。

图 3.12(b)是把一个横向为时间、纵向为产品的报表变成一个横向仍为时间而纵向旋转为地区的报表。

3.4.2 广义 OLAP 功能

OLAP 的切片、切块、旋转与钻取等基本操作是最基本的展示数据，也是获取数据信息的手段。从广义上讲，任何有助于辅助用户理解数据的技术或者操作都可以作为 OLAP 功能，这些有别于基本 OLAP 的功能称为广义 OLAP 功能。

1. 基本代理操作

“代理”是一些智能性代理，当系统处于某种特殊状态时提醒分析员。

(1) 示警报告

定义一些条件，一旦条件满足，系统会提醒分析员去做分析。如每日报告完成或月订货完成等，通知分析员做分析。

(2) 时间报告

按日历和时钟提醒分析员。

(3) 异常报告

当超出边界条件时提醒分析员。如销售情况已超出预定义阈值的上限或下限时提醒分析员。

2. 数据分析模型

E. F. Codd 认为，以前的数据分析主要集中在静态数据值的相互比较上。有了 OLAP 后，可以进行动态数据分析，需要建立企业数据分析模型。E. F. Codd 将数据分析模型分为 4 类模型：绝对模型(categorical model)、解释模型(exegetical model)、思考模型(contem-

plative model）和公式模型（formulaic model）。

（1）绝对模型

它属于静态数据分析，通过比较历史数据值或行为来描述过去发生的事实。该模型查询比较简单，综合路径是预先定义好的，用户交互少。

（2）解释模型

它也属于静态数据分析，分析人员利用系统已有的多层次的综合路径层层细化，找出事实发生的原因。

（3）思考模型

它属于动态数据分析，旨在说明在一维或多维上引入一组具体变量或参数后将会发生什么。分析人员在引入确定的变量或公式关系时，必须创建大量的综合路径。

（4）公式模型

它的动态数据分析能力更高，该模型表示在多个维上，需要引入哪些变量或参数，以及引入后所产生的结果。

下面通过一个实例进行说明。

一家百货公司在建立了自己的数据仓库之后，希望构造一个 OLAP 系统辅助决策。决策者最关心的一个问题是如何最大限度地扩大商品的销售量，因而希望能尽可能找出与销售量相关的因素，从而采取相应的促销手段。但是能获得多大的帮助取决于采用何种分析模型。

绝对模型只能对历史数据进行比较，并且利用回归分析等一些分析方法得出趋势信息。能回答诸如“某种商品今年的销售情况与以往相比有怎样的变化？今后的趋势怎样？”等类问题。

解释模型能够在当前多维视图的基础上找出事件发生的原因。例如，该公司按时间、地区、商品及销售渠道建立了多维数据库，假设今年销售量下降，那么解释模型应当能找出原因，即销售量下降与时间、地区、商品及销售渠道四者中的何种因素有关。

思考模型在决策者的参与下，找出关键变量。例如该公司决策者为了了解某商品的销售量是否与顾客的年龄有关，引入了行变量——年龄，即在当前的多维视图上增加了顾客的年龄维。解释模型就能分析出年龄的引入是否必要，即商品销售与顾客年龄有关或无关。

公式模型自动完成上述变量引入工作，从而最终找出与销量有关的全部因素，并给出引入后的结果。

可以看出，这 4 种模型，从描述基本事实到寻找原因，从代入变量值进行预测到寻找关键变量，一个比一个深入。

E. F. Codd 认为 OLAP 是因企业动态分析而产生的，其功能是创建、操作、激活及综合来自解释模型、思考模型及公式模型中的信息。它可以识别变量间的新的或不可预测的关联，通过创建大量的维（综合路径）及指出维间计算条件、表达式来处理大量数据，获得辅助决策信息。

3. 商业分析模型

利用数据仓库中的数据进行商业分析需要建立一系列模型，用于提高决策支持能力。

具体的商业分析模型有：

(1) 分销渠道的分析模型

通过客户、渠道、产品或服务三者之间的关系，了解客户的购买行为、客户和渠道对业务收入的贡献、哪些客户比较喜好由什么渠道在何时和银行打交道、目前的分销渠道的服务能力如何、需要增加哪些分销渠道才能达到预期的服务水平。

为此，银行需要建立客户购买倾向模型和渠道喜好模型等。

(2) 客户利润贡献度模型

通过该模型能了解每一位客户对银行的总利润贡献度，银行可以依客户的利润贡献度安排合适的分销渠道，提供服务和销售，知道哪些利润高的客户需要留住，采用什么方法留住客户，交叉销售改善客户的利润贡献度，哪些客户应该争取，完成个性化服务。另外，银行可以模拟和预测新产品对银行的利润贡献度，或者新政策对银行将产生什么样的财务影响，或者客户流失或留住对银行的整体利润的影响。

(3) 客户关系(信用)优化模型

银行从客户的每一笔交易中知道客户需要什么产品或服务，例如，定期存款是希望退休养老使用；申请信用卡需要现金消费；询问放贷利息需要住房贷款等，这些都是银行提供产品或服务最好的时机。银行需要将每个账号每天发生的交易明细以实时或定时方式加载到数据仓库中，校对客户行为的变化。当有上述变化时，通过模型计算，主动地与客户沟通并进行交叉销售，以达到留住客户和增加利润的目标。

(4) 风险评估模型

模拟风险和利润间的关系，建立风险评估的数学模型，在满足高利润、低风险客户需求的前提下，达到银行收益的极大化。

银行通过以上模型建立以客户为中心的数据仓库决策支持系统，才能真正实现个性化服务，提高银行竞争优势。

3.4.3 多维数据分析实例

假设有一个五维数据模型，5 个维分别为：商店，方案，部门，时间，销售。下面进行实例分析。

1. 多维数据存储

指定“商店＝ALL(广州所有商店)，方案＝现有”情况的三维表(行为部门，列为时间和销售量)，如表 3.10 所示。

表 3.10 中无括号数为增长率，有括号表示下降率，下同。

对于汽车部门出现的奇怪现象，销售下降了 13.2%，而利润却增加了 21.4%，此时进行向下钻取。

表 3.10　指定商店、方案后的三维表

商店＝All　方案＝现有

项目	2004 年		2005 年		增长率/%	
	销售量	利润增长/%	销售量	利润增长/%	销售量	利润增长
服装	234 670	27.2	381 102	21.5	62.4	(20.0)
家具	62 548	33.8	66 005	31.1	5.6	(8.0)
汽车	375 098	22.4	325 402	27.2	(13.2)	21.4
所有其他	202 388	21.3	306 677	21.7	50.7	1.9

2. 向下钻取

对汽车部门向下钻取出具体项目(维修、附件、音乐)的销售情况和利润增长情况，见表 3.11 所示。

表 3.11　下钻数据

项目	2004 年		2005 年		增长率/%	
	销售	利润增长/%	销售	利润增长/%	销售	利润增长
汽车	375 098	22.4	325 402	27.2	(13.2)	21.4
维修	195 051	14.2	180 786	15.0	(7.3)	5.6
附件	116 280	43.9	122 545	47.5	5.3	8.2
音乐	63 767	8.2	22 071	14.2	(63.4)	7.3

3. 切片表

切片(slice)操作是除去一些列或行不显示，如对表 3.10 的切片为表 3.12 所示。

表 3.12　切片表

商店＝All　方案＝现有

项目	2005 年
	销售量
服装	381 102
家具	66 005
汽车	325 402
所有其他	306 677

4. 旋转表

将方案维加入到销售维中。方案维有 3 种情况：现有、计划、最新预测，这次旋转操作得到 2005 年的表 3.10 中方案维的成员有：现有、计划、差量、差量(%)，得到旋转表如表3.13 所示。

表 3.13　旋转表

商店＝ All 　方案＝ 现有

项目	2005 年			
	销售量			
	现有	计划	差量	差量/%
服装	381 102	350 000	31.1	8.9
家具	66 005	69 000	(2995)	(4.3)
汽车	325 402	300 000	25 402	8.5
所有其他	306 677	350 000	(43 323)	12.7

3.5　OLAP 结构与分析工具

3.5.1　OLAP 结构

OLAP 的实现是基于客户/服务器(C/S)模式的。

1. OLAP 逻辑结构

OLAP 逻辑结构由 OLAP 视图和数据存储两部分构成，如图 3.13 所示。

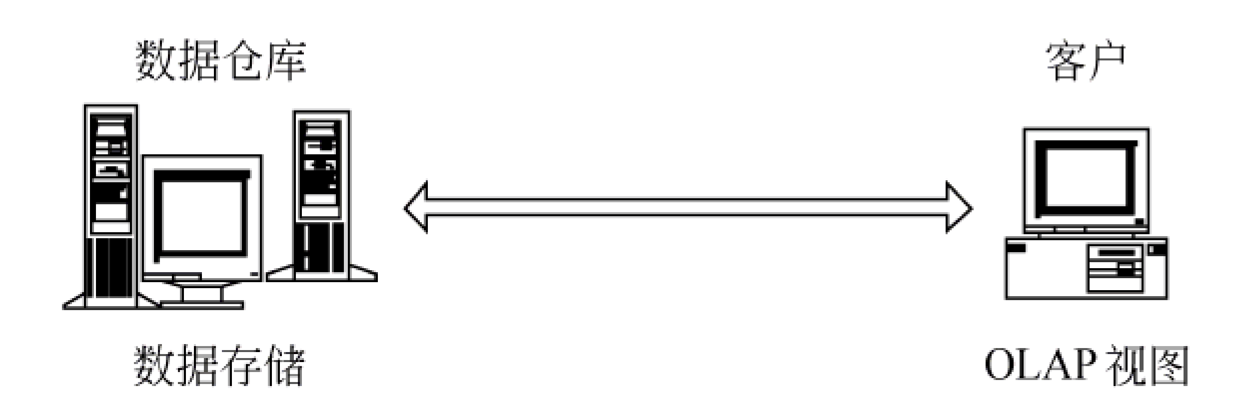


图 3.13　OLAP 逻辑结构

(1) OLAP 视图：对于用户来说它是数据仓库或数据集市中数据的多维逻辑表示，不管数据怎样存储和存储在何处。

(2) 数据存储：要求选择数据实际存储的方式和实际存储的位置，两种常用的选择是多维数据存储和关系数据存储。

2. OLAP 物理结构

物理结构包括基于数据存储的两种方式：多维数据存储和关系数据存储。

多维数据存储主要有两种选择：多维数据存储于客户端或 OLAP 服务器。在第一种情况，多维数据存储于客户端，数据分析也在客户端，这样形成了“胖”客户端。这种两层客户/服务器(C/S)的物理结构，如图 3.14 所示。



图 3.14　OLAP 的两层 C/S 物理结构

在第二种情况，多维数据存储放在 OLAP 服务器中，抽取数据仓库中的数据，然后将其转换成多维数据结构，并把 OLAP 服务传给客户端，这时客户端就变成“瘦”客户端，这是一种经典的三层客户/服务器物理结构，如图 3.15 所示。



图 3.15　OLAP 的三层 C/S 物理结构

3.5.2　OLAP 的 Web 结构

当使用 Web 结构组织 OLAP 应用时，其组织结构如图 3.16 所示。

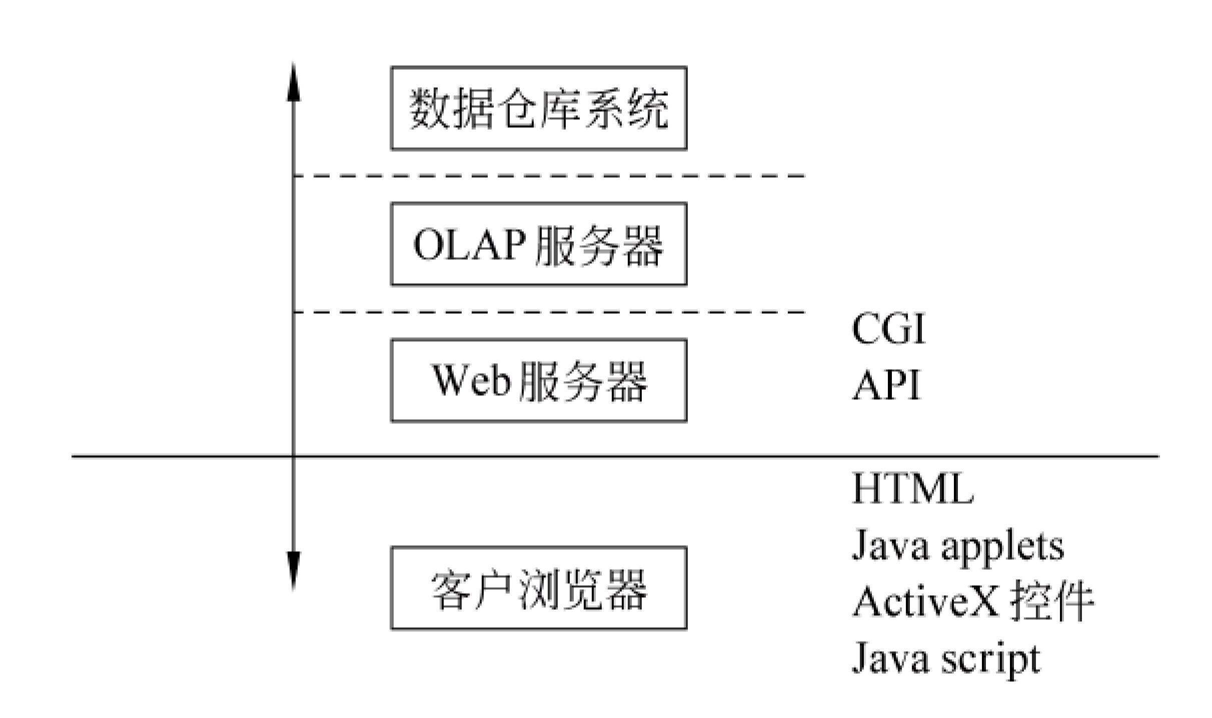


图 3.16　基于 Web 的 OLAP 结构图

Web 服务器负责完成浏览器与 OLAP 服务器、数据仓库系统之间的通信连接。一般来说，Web 服务器通过使用 CGI 脚本、Web 服务器 API、应用 API 和数据库 API 等，管理浏览器的通信。在浏览器端，则需要组织 HTML、Java applets、ActiveX 控件、Java script 来完成与用户的交互界面和控制。

在实现基于 Web 的 OLAP 应用时，往往采用自顶向下的设计。首先要确定用户如何在浏览器中得到报表信息，然后再给出一系列的过程完成基于浏览器的 OLAP 操作。这些过程应当包括发布信息，提供 HTTP 对数据库或应用服务器的动态数据请求，设计支持分析功能的界面。

Web 服务器要为发布准备和创建信息，管理员必须生成一些报表和图表为查询作准备。在客户端基于 Web 浏览器的 OLAP 报表被分为几个层次，其中包括没有分析功能的静态报表和能够进行分析的 OLAP 应用界面，如维的旋转、数据的钻取等。另外，特别要提到的是，标准的 HTML 界面缺乏操作的灵活与方便。Java applets 和 ActiveX 控件的使用会大大提高用户界面的友好程度，用户通过简单的拖放操作就可以完成“旋转”、“钻取”的操作。

大体上来说，实现基于 Web 的 OLAP 有 3 种方法：

- 静态的 HTML 报表；
- 通过 HTML 模板、元数据动态生成报表；
- 使用 Java 或 ActiveX 的改进方式。

1. 静态方法

这种方法使用的数据是“脱机”方式制作的。也就是说，制作者根据 HTML 模板和某个时刻的数据为 Web 服务器制作大量的 HTML 页面。这些 HTML 页面是静态的，代表某个时间点的数据，用户得到的是一些固定的报表。当用户要求完成 OLAP 操作时，Web 服务器能够提供一种“仿真”方式。可以预先制作一些 HTML 页面，使用超文本的跳转方式组织起来，使用户在使用上感觉是在“钻取”或“旋转”。

这种方法有 3 个特点：

(1) 用户所见到的数据是脱机的，OLAP 功能的完成是一种仿真模式。

(2) 由于 HTML 页面是事先制作好的，数据又与数据库分离，响应速度快。

(3) 尽管 OLAP 应用实时性要求不高，但仍需要 Web 调度为服务器定期制作和更换 HTML 页面。

可以看出，这种方法实现简单，可以作为系统试运行和用户培训的先验系统。

2. 动态方法

这是一种通用的方法，在用户有服务请求时，服务器根据 HTML 模板和数据库中的数据动态地生成 HTML 页面。与前一种方法不同的是，Web 服务器只存放 HTML 模板和元数据，这里元数据可以告诉 Web 服务器在哪里和怎样得到数据。

HTML 模板在服务器上的存放也有两种形式：一种是标准的 Web 形式，事先生成一些模板，在用户请求时将数据与模板结合，返回客户端。另一种模式是厂商为方便用户设计 HTML 所采用的特殊方法。这种方法将元数据与模板存放在统一的数据库格式下，用户在设计这种网页时，通过可视化的界面完成设计，设计工具根据这个设计将所有的设计信息，包括模板样式和元数据存储到数据库。当用户请求服务器服务时，服务器通过特殊的解释工具解释数据库中的模板与元数据信息组织 HTML 页面，返回客户端。

由于服务器对 HTML 模板与元数据的存储独立于客户端，客户端接收的是 HTML 页面，因此这种方法是一种方便的组织方式，可以使客户端跨平台使用。这种方便的特性主要来自 CGI 的工作方式(因为 CGI 在服务器启动程序得到一个 HTML 页面，然后向客户端返回)，而使用 Web 服务器，API 则往往只能在相对固定的平台中使用。

这种方法是基于 Web 的 OLAP 系统的主流组织方法，其特点主要有：

(1) HTML 模板与元数据存储在 Web 服务器上，动态形成 HTML 页面，实时性好。

(2) 响应时间没有第一种方法快，有时甚至很慢。

(3) 服务器具有跨平台的独立性。

3. 改进的方法

这种方法是第二种方法的改进型，这种方法引进了 Java 和 ActiveX 技术，使得用户界面更加友好，使用更加方便。

具体来说，这种方法的实现有两种途径。第一种途径是在服务器生成二进制的数据文件，服务器将这些数据传送到客户端，这些数据与相应的控件联系，控件的属性可以得到这些文件的标识。控件可以告知浏览器下载这个数据文件，再将这个文件与一个数据构件联系，以后对于数据的 OLAP 操作(旋转、钻取)都由这个数据构件提供，不再向服务器请求服务。第二种途径是让数据构件向服务器请求用户要求的数据，构件得到的是开放的 HTTP 流，在得到以后将这些数据流分解，显示到交互界面上。

3.5.3 OLAP 工具及评价

随着 OLAP 技术研究的深入，目前许多公司已经推出了相应的 OLAP 支持工具，如 Oracle、IBM、Business Object、SAS、NCR 等。

对于功能特点不同、应用领域不同以及技术特性不同的 OLAP 产品，本节给出了对 OLAP 分析工具的评价指标。

OLAP 服务器和工具可以按以下 5 个方面进行评价：特征和功能、访问性能、OLAP 服务引擎、管理以及全局结构。用户可以从这 5 个方面分析市场上的 OLAP 产品，也可以把它们作为应用系统中 OLAP 需求分析指标。

1. 特征和功能

OLAP 是一种分析处理技术，通过计算公式和转换规则从现有的数据中生成新的信息，并予以显示。OLAP 服务器和工具应能完成以下功能。

- 支持多维和维中的层次；
- 沿单个维或沿一组所选维来聚集、概括、预计算和导出数据；
- 相对一个维或一组选中的维提供计算逻辑、公式和分析过程；
- 支持分析模型的概念：分析模型是一组选中的维及维的元素、计算逻辑、公式、分析过程、聚集数据、概括数据和导出数据等；
- 提供丰富的库函数；
- 提供强大的计算和比较分析能力，例如：分级、比较、归类百分比、极大值、极小值、平均值、按时期的比较等；
- 进行跨维计算，例如在面向电子表格的应用程序中进行行级别的计算等；
- 提供时间相关的智能，例如，按日期划分的年，跨越给定时间段的日历、当前时期、财政的和内部的日历等；
- 从一个维到另一个维进行转换，在合并或获取数据后特别有用；

• 导航并分析，采用沿单个或多个维的轴以及交叉表等进行细剖和浏览。

这些操作应满足用户分析时的要求，分析过程应是平滑的、连贯的。

2. 访问性能

OLAP 的访问服务应提供多种选择，潜在的选择应包括以下内容。

(1) 电子表格：商业用户至少应能将 OLAP 数据加载至他们的电子表格工具，因其他的分析和报表可能要用到这些数据；

(2) 私用客户工具：在提供一个特定的应用时，例如预算，用户总是希望尽快开始分析处理。其功能是否丰富、是否能满足用户要求，是关键的标准；

(3) 客户的"立体导航器"：它们是源于第三方的工具，但存在与 OLAP 服务器的接口，访问接口应完成以下功能：

• 访问并抽取基于层次、模型、时间和其他所选维的数据子集；
• 用单个抽取请求访问多个层次；
• 了解聚集和概括数据及其划分方式和索引，以便生成适当的查询；
• 在访问关系数据存储时，优化特定的关系数据库，包括关系数据库中 SQL 的扩充。

3. OLAP 服务引擎

无论是采用多维存储还是关系存储，OLAP 服务引擎都应满足分析模型及应用在功能、规模和技术特征上的要求。技术特征的需求依赖于分析模型和希望采用的方式。其中一些特征如下：

• 读写功能：用于交互式预测和预算的应用程序。
• 多用户写操作：支持按工作组进行的多维分析。OLAP 多用户写操作比直接写关系数据时遇到的问题要多得多。OLAP 修改或写请求不仅仅只是考虑表中的一行，可能需要重计算派生的和经计算得到的信息，这些信息将影响多个维和维的层次。
• 多数据库：如果每一个 OLAP 应用程序都有一数据库，它可能需要在数据库间进行交互的机制，因为一个数据库中产生的数据可能要输入其他数据库。例如，财务 OLAP 应用程序需要源于销售 OLAP 应用程序的收入信息，以便建立损益报表并把实际情况与预测情况进行比较。
• 数据类型的范围：包括数字、时间/日历、描述(用于显示和报表)等。生成更多图像数据类型，可以增强动态显示和执行报表的功能，有利于加强复杂分析的表达。

4. 管理

初始准备、设置和连续操作需要管理功能，它包括：

• 定义维分析模型；
• 生成并维护元数据存储；
• 访问控制和基于使用的权限，此处集中于用户需要做什么，以及谁可以访问分析模型及数据；
• 从数据仓库或数据集市加载分析模型；

- 协调行为至可接受的层次级别，并可进行不受干扰的分析；
- 为增强数据库的性能，或者为了修改维模型，或是为了修改数据，重新组织数据库；
- 管理系统的各个部分，包括中间件，参照结构提供的方式来了解系统管理任务的范围；
- 把数据传送给客户，以便进一步分析或作本地分析。

5. 全局结构

从全局结构上看，OLAP 是采用关系的还是多维数据存储，不能简单地作出选择。各种应用需求才是判断所做决策是否正确的标准。

当今的趋势是组合 OLAP 服务器前端和关系存储的后端来提供 OLAP 服务。此配置是将综合数据嵌入多维存储而放在前端的 OLAP 服务器中，后端的细节级的详细数据仍采用关系存储。事实上，有些企业刚开始时使用关系存储，需要时才再添加多维存储。

在此种结构的配置中，需要对经常访问的信息和经常使用的查询做预计算、概括、聚集，然后将其存储在 OLAP 服务器的多维数据存储中，这可在刚开始从数据仓库（或数据集市）中加载分析模型时进行。复杂的或集约计算的查询，以及复杂的基于计算的数据也可做预处理并存储，这可加快操作速度。

对于从少量的维元素中计算出的信息或数据，如果不需经常访问它们，则只在收到查询时才计算。这些不常被访问的数据甚至不必存于多维数据存储中，只在需要的时候，由 OLAP 服务器从关系数据存储中检索它们。

利用管理功能可以在多维数据存储中存储数据或结果，以便处理后继请求，这些结果是那些不常被访问的查询产生的，增强了总体性能，并只在需要时才增加存储。

此配置也支持剖析细节数据。多维数据存储不提供细节数据，通过生成请求来检索源于关系存储的细节数据。

重要的问题是应保持各种应用的目标并牢记按用户的观点来处理数据。一个良好的 OLAP 方案应在以上所讨论的 5 个方面以及生存期的各种开销（如初始获取和安装、训练、维护和运行）间达成适当的平衡。

习　题

1. 联机分析处理（OLAP）的简单定义是什么？它体现的特征是什么？
2. OLAP 准则中主要准则有哪些？
3. 什么是维？关系数据库是二维数据吗？如何理解多维数据？
4. MDDB 与 RDBMS 有什么不同？说明各自特点。
5. 比较 ROLAP 与 MOLAP 在数据存储、技术及特点上的不同。
6. HOLAP 数据模型的特点是什么？
7. 举例说明多维数据显示的两种不同方法。
8. 举例说明多维类型结构（MTS）。
9. 举例说明四维数据显示。

10. 举例说明六维数据显示。

11. 多维数据显示的经验规则是什么?

12. 举例说明 OLAP 的多维数据分析的切片操作。

13. 举例说明 OLAP 的多维数据分析的切块操作。

14. 举例说明 OLAP 的多维数据分析的钻取功能。思考在计算机中如何实现这种功能。

15. 说明 4 种不同的数据分析模型的差别。

16. 举例说明 4 种数据分析模型的应用,以及如何提高 OLAP 分析能力。

17. 解释 OLAP 逻辑结构。

18. 解释 OLAP 两种物理结构,并说明它与数据仓库的运行结构关系。

19. 解释 OLAP 的 Web 结构。

20. OLAP 分析工具应该具有的功能有哪些?

21. OLAP 服务引擎应满足的技术特征有哪些?

第4章　数据仓库设计与开发

4.1　数据仓库分析与设计

数据仓库分析与设计由需求分析、概念模型设计、逻辑模型设计与物理模型设计4个部分组成。

4.1.1　需求分析

数据仓库是一个向用户提供战略信息的环境，从而为用户提供决策支持。数据仓库不同于现行的事务处理系统（或称操作型系统），事务处理系统完成每日的业务运行，对于用户所需的功能、信息内容、使用方式，系统都有清楚的定义。数据仓库不能清楚地定义用户的需求，即不能准确定义用户真正想从数据仓库中得到哪些信息，也不能说明如何使用和处理这些信息。但是，用户可以指出哪些是重要的衡量指标，如何将各种信息综合起来为战略决策服务。

例如，市场部经理感兴趣的是每个月、某个地区、按照销售部门、参照历史数据和计划数据，了解新产品创造多少利润。销售经理需要按照产品种类，每天、每星期、每月进行汇总，按照销售地区或按销售渠道进行统计。财务经理在制定费用列表时，要与预算比较，按照每月、每季度和每年，按照预算资金定义，按照地区，对全公司进行汇总统计。

数据仓库的需求分析是数据仓库设计的基础。需求分析的任务是通过详细调查现实世界要处理的对象（企业、部门、用户等），充分了解原系统（人工系统或计算机系统）的工作概况，明确用户的各种需求（包括当前的需求和长远的需求），为设计数据仓库服务。概括地说，需求分析要明确用哪些数据经过分析来实现用户的决策支持需求。

数据仓库用户包括高层主管、部门经理、IT专业人员等。通过对用户的调查，对数据仓库系统需要确定的问题为：

1. 主题域

（1）明确对于决策分析最有价值的主题领域有哪些？

（2）每个主题域的商业维度是哪些？每个维度的粒度层次有哪些？

（3）制定决策的商业分区是什么？

（4）不同地区需要哪些信息来制定决策？

（5）对哪个区域提供特定的商品和服务？

2. 支持决策的数据来源

（1）哪些源数据（操作型）与商业主题有关？

（2）在已有报表和在线查询中得到什么样的信息？
（3）提供决策支持的细节程度是怎样的？

3. 数据仓库的成功标准和关键性能指标

（1）衡量数据仓库成功的标准是什么？
（2）有哪些关键的性能指标？如何监控？
（3）对数据仓库的期望是什么？
（4）对数据仓库的预期用途有哪些？
（5）对计划中的数据仓库的考虑要点是什么？

4. 数据量与更新频率

（1）数据仓库的总数据量有多少？
（2）决策支持所需的数据更新频率是多少？时间间隔是多长？
（3）每种决策分析与不同时间的标准对比如何？
（4）数据仓库中的信息需求的时间界限是什么？

通过需求分析，明确决策支持所需要的数据，包括如下内容：

1. 数据源

建立数据仓库需要使用源系统的数据，从这些源系统中收集、合并和整合数据，正确地转换这些数据，装入到数据仓库中。

数据源中的数据包括：
（1）可用的数据源；
（2）数据源的数据结构；
（3）数据源的位置；
（4）数据源的计算机环境；
（5）数据抽取过程；
（6）可用的历史数据。

2. 数据转换

数据仓库中的数据是为决策分析服务的，不同于源系统的数据为业务处理服务。这样需要决定如何正确地将这些源数据转换成适合数据仓库存储的数据类型。

在需求分析文档中要包括数据转换的细节，不但要明确从什么地方得到数据，并描述在将数据载入数据仓库之前的合并、转化和分拆的过程。

3. 数据存储

通过对用户的采访，会发现数据仓库所需要的数据的详细程度，包括足够的关于存储需求的信息，估计数据仓库需要多少历史和存档数据。

4. 决策分析

需求分析文档应该包括用户决策分析的需求,即:

(1) 向下层钻取分析;

(2) 向上层钻取分析;

(3) 横向钻取分析;

(4) 切片分析;

(5) 特别查询报表。

4.1.2 概念模型设计

将需求分析过程中得到的用户需求抽象为信息结构,即为概念模型。它是从客观世界到计算机世界的一个中间层次。

概念模型的特点是:

(1) 能真实反映现实世界,能满足用户对数据的分析,达到决策支持的要求,是现实世界的一个真实模型。

(2) 易于理解,便利和用户交换意见,在用户的参与下,能有效地完成对数据仓库的成功设计。

(3) 易于更改,当用户需求发生变化时,容易对概念模型修改和扩充。

(4) 易于向数据仓库的数据模型(星型模型)转换。

概念模型最常用的表示方法是实体-关系法(E-R 法),这种方法用 E-R 图作为它的描述工具。E-R 图描述的是实体以及实体之间的联系,用长方形表示实体,在数据仓库中就表示主题,在框内写上主题名,椭圆形表示主题的属性,并用无向边把主题与其属性连接起来;用菱形表示主题之间的联系,菱形框内写上联系的名字,用无向边把菱形分别与有关的主题连接,在无向边旁标上联系的类型。若主题之间的联系也具有属性,则把属性和菱形也用无向边连接上。

由于 E-R 图具有良好的可操作性,形式简单,易于理解,便于与用户交流,对客观世界的描述能力也较强,在数据库设计方面得到广泛的应用。因为目前的数据仓库一般建立在关系数据库的基础之上,与数据库的概念模型相一致,采用 E-R 图作为数据仓库的概念模型仍然是较为适合的。

通过一个例子来说明数据仓库的概念模型的设计,有两个主题:商品和客户。主题也是实体。

商品有如下属性组:

- 商品的固有信息(商品号、商品名、类别、价格等);
- 商品库存信息(商品号、库房号、库存量、日期等);
- 商品销售信息(商品号、客户号、售价、销售日期、销售量等);
- 其他信息等。

客户有如下属性组:

- 客户固有信息(客户号、客户名、性别、年龄、文化程度、住址、电话等);

• 客户购物信息(客户号、商品号、售价、购买日期、购买量等)。

其中商品的销售信息与用户的购物信息是一致的,它们是两个主题之间的联系。

将两个主题的概念模型用 E-R 图画出,如图 4.1 所示。

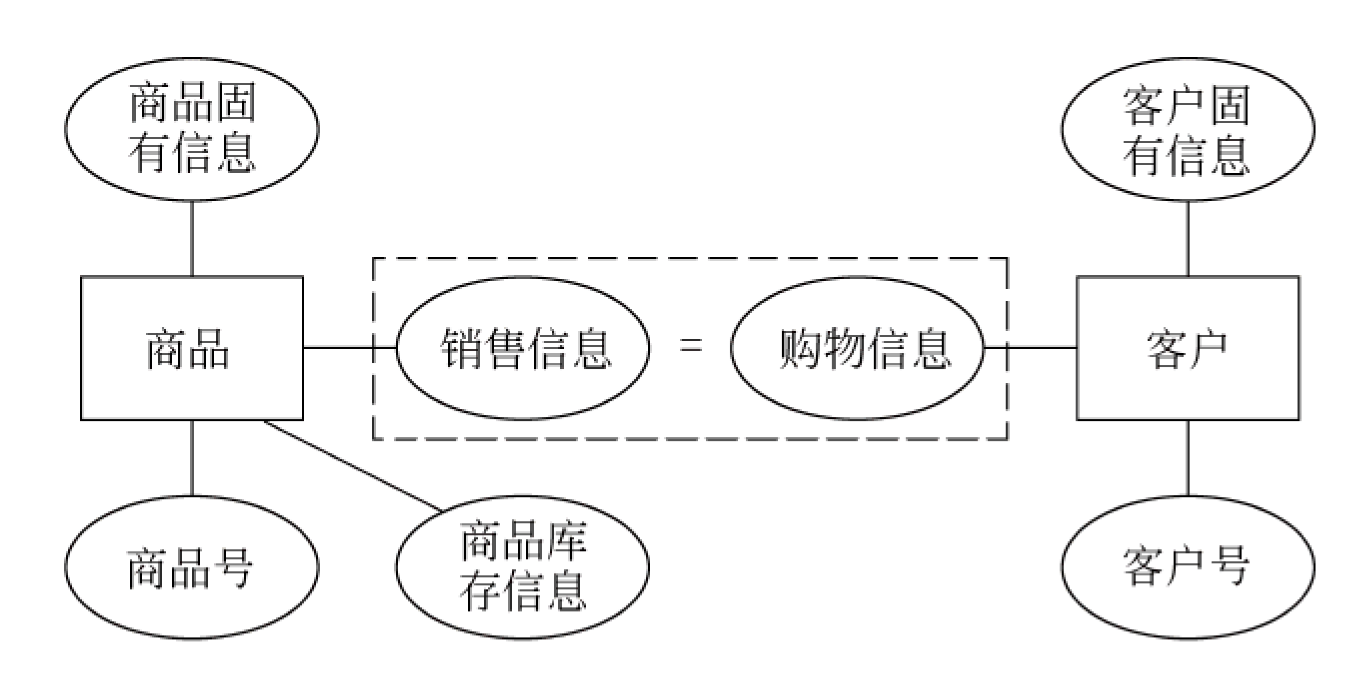


图 4.1　商品与客户两主题的概念模型

4.1.3　逻辑模型设计

逻辑模型设计是把概念模型设计好的 E-R 图转换成计算机所支持的数据模型。数据仓库在计算机中的数据模型是星型模型。这样数据仓库的逻辑模型设计主要是将用 E-R 图表示的概念模型转换成星型模型。

数据仓库逻辑模型设计的主要工作为:

(1) 主题域进行概念模型(E-R 图)到逻辑模型(星型模型)的转换;

(2) 粒度层次划分;

(3) 关系模式定义;

(4) 定义记录系统。

1. 主题域进行概念模型到逻辑模型的转换

在概念模型设计中,可能确定了多个主题域。但是,数据仓库的设计一般是从一个或几个主题逐步完成的。选择第一个主题域要足够大,使该主题能完成围绕该主题的决策分析需要。但要足够精练,便于开发和较快实施。

例如,概念模型设计时,确定了"商品"和"客户"两个主题。其中"商品"对于商场来说是更基本的业务对象。商品的业务有销售、采购、库存等,其中商品销售是最主要的业务,是进行决策分析的最主要方面。因而,"商品"主题比"客户"主题更重要。

星型模型的设计步骤如下:

(1) 确定决策分析需求

数据仓库是面向决策分析的,决策需求是建立多维数据模型的依据。如分析销售额趋势、对比商品销售量、促销手段对销售的影响等。

(2) 从需求中识别出事实

在决策主题确定的情况下,选择或设计反映决策主题业务的表,如在"商品"主题中,以"销售业务"作为事实表。

(3) 确定维

确定影响事实的各种因素，销售业务的维包括商店、地区、部门、城市、时间、商品、促销等，如图 4.2 所示。

(4) 确定数据汇总的水平

在数据仓库中的数据包括汇总的数据。数据仓库中对数据不同粒度的综合，形成了多层次的数据结构。例如，对于时间维，可以以“年”、“月”或者“日”等不同水平进行汇总。

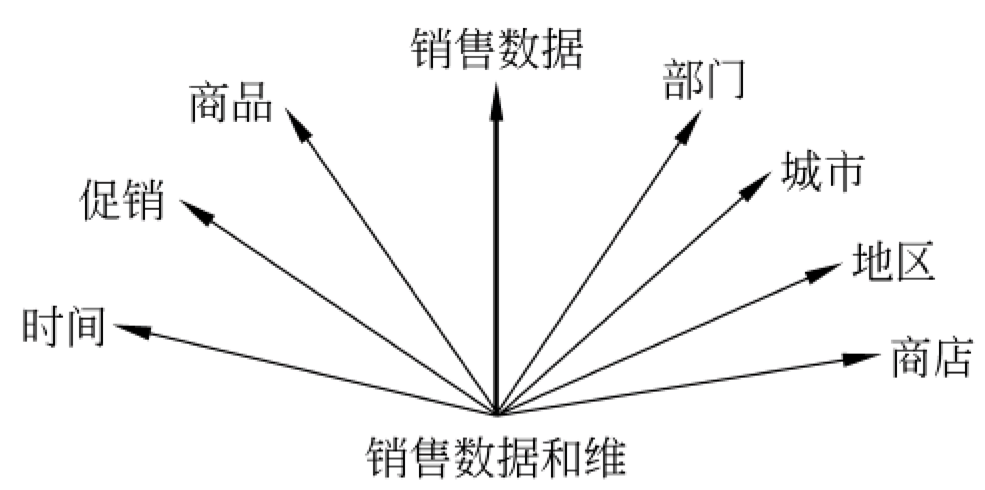


图 4.2　销售业务的多维数据

(5) 设计事实表和维表

设计事实表和维表的具体属性。在事实表中应该记录哪些属性是由维表的数量决定的。一般来说，与事实表相关的维表的数量应该适中，太少的维表会影响查询的质量，用户得不到需要的数据，太多的维表又会影响查询的速度。

(6) 按使用的 DBMS 和分析用户工具，证实设计方案的有效性

根据系统使用的 DBMS，确定事实表和维表的具体实现。由于不同的 DBMS 对数据存储有不同的要求，因此设计方案是否有效，还要放在 DBMS 中进行检验。

(7) 随着需求变化修改设计方案

随着应用需求的变化，整个数据仓库的数据模式也可能会发生变化。因此在设计之初，充分考虑数据模型的可修改性，可以节省系统维护的代价。

从概念模型的 E-R 图转换成逻辑模型的星型模型，实例说明如下：

(1) 业务数据的 E-R 图

见图 4.3。

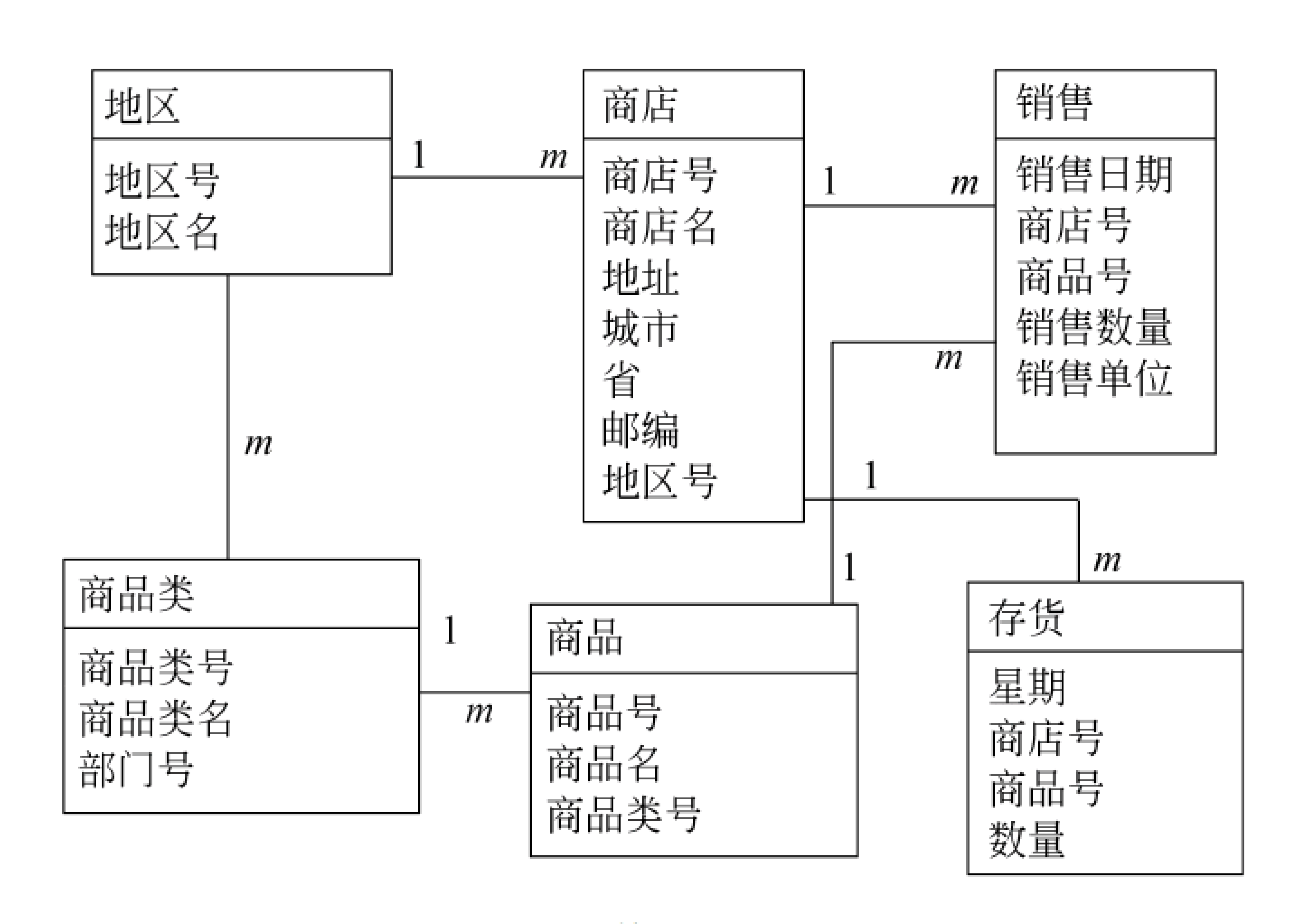


图 4.3　实体关系(E-R)图

(2) E-R 图向多维表的转换

该问题的多维表模型中，商品维包括部门、商品和商品大类，地区维包括地区和商店，忽略存货，而只注意销售事实。在 E-R 图中不出现时间，在多维模型中增加时间维，如图 4.4 所示。

在多维模型中，实体与维之间建立映射关系，联系多个实体的实体就成为事实，此处销

售实体作为事实，其他实体作为维，然后用维关键字将它转换为星型模型，如图 4.5 所示。

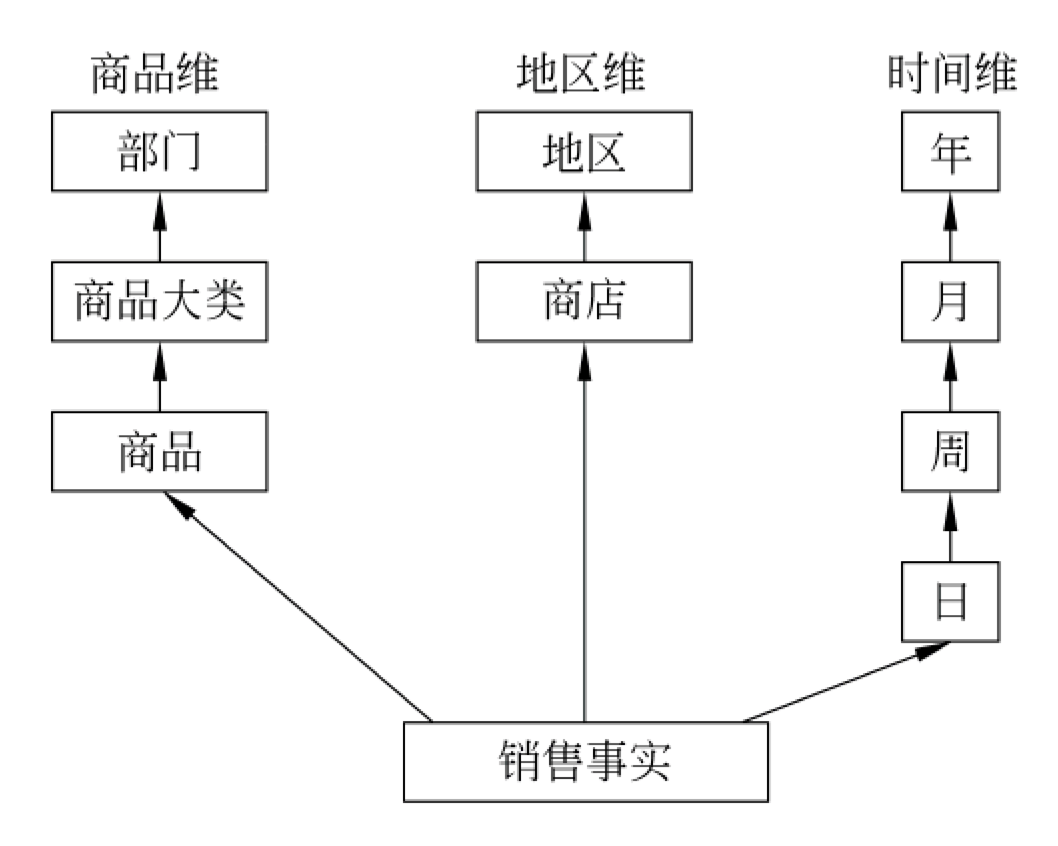


图 4.4　E-R 图向多维模型的转换

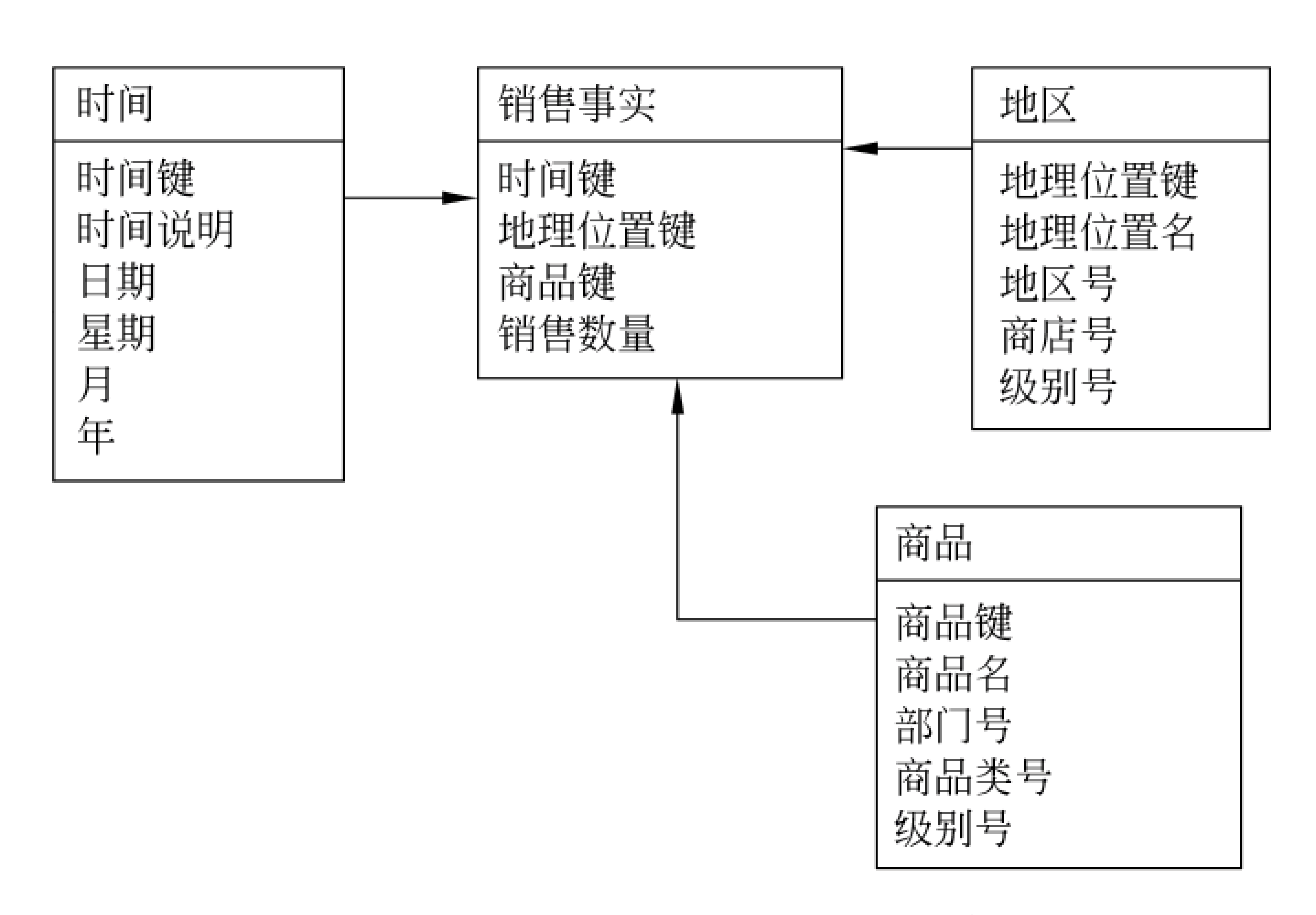


图 4.5　利用维关键字制定的星型模型

其中地区维是综合了“地区”和“商店”两个实体，它们有一个层次的差别。将“商店”作为 1 级，“地区”作为 2 级，该维的关系表如表 4.1 所示。

表 4.1　地区维关系表

地理位置键	地理位置名	地区号	商店号	级别号
100	东北地区	1		2
105	中西部	2		2
110	中南地区	3		2
115	沈阳	1	2204	1
120	西安	2	2349	1
125	长春	1	2542	1
130	广州	3	2211	1

在各维中，只有部门、商品类、地区、商店的编号没有具体的说明，为了打印报表将增加这些编号的名称说明，即部门名、商店名等，在维表中增加这些说明，即修改该星型模型，如

图 4.6 所示。

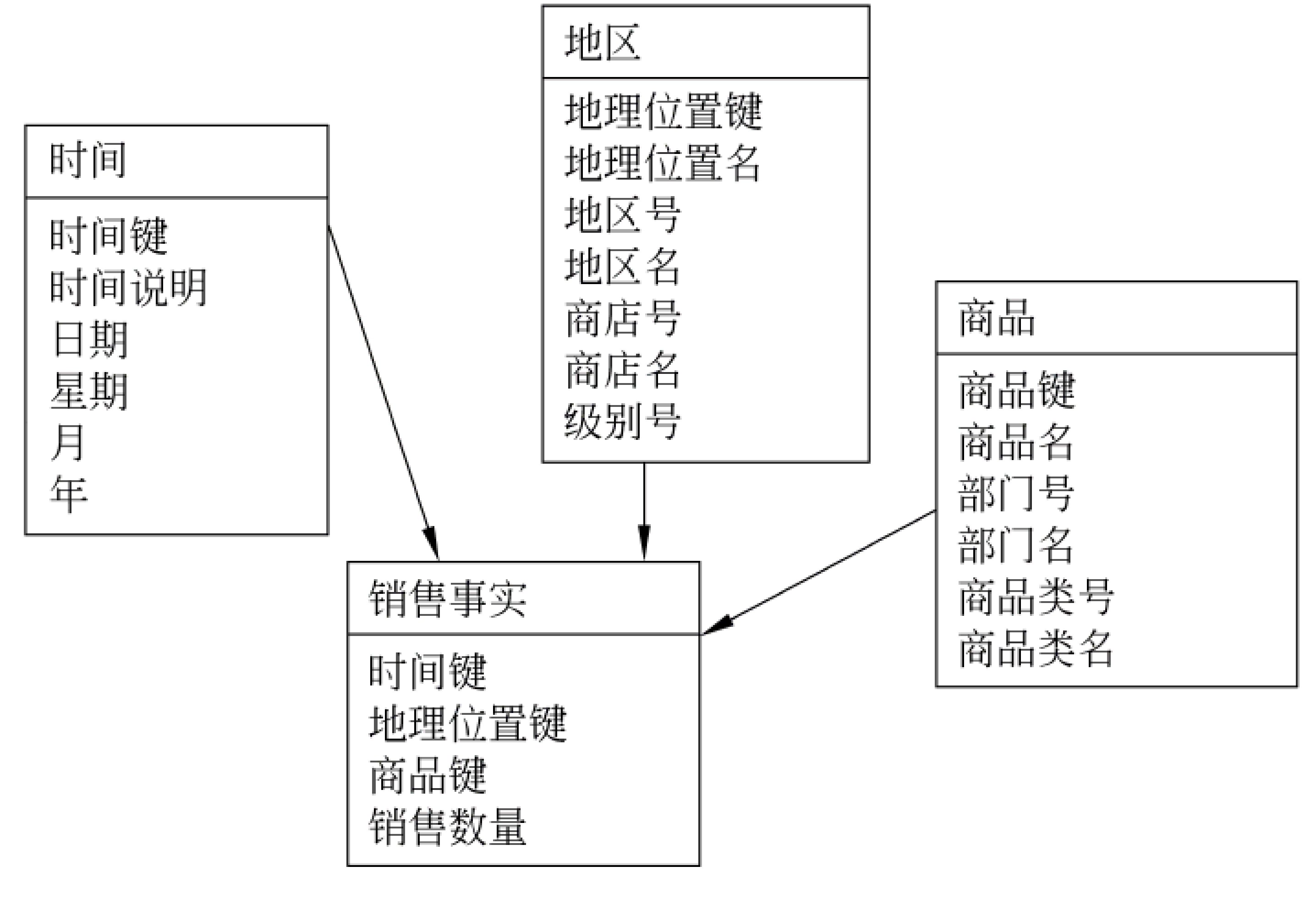


图 4.6 修改后的星型模型

2. 粒度层次划分

所谓粒度是指数据仓库中数据单元的详细程度和级别。数据越详细,粒度越小,层次级别就越低;数据综合度越高,粒度越大,层次级别就越高。在传统的操作型系统中,对数据的处理和操作都是在详细数据级别上的,即最低级的粒度。但是在数据仓库环境中主要是分析型处理,粒度的划分将直接影响数据仓库中的数据量以及所适合的查询类型。一般需要将数据划分为详细数据、轻度综合、高度综合三级或更多级粒度。不同粒度级别的数据用于不同类型的分析处理。粒度的划分是数据仓库设计工作的一项重要内容,粒度划分是否适当是影响数据仓库性能的一个重要方面。

进行粒度划分,首先要确定所有在数据仓库中建立的表,然后估计每个表的大约行数。在这里只能估计一个上下限。需要明确的是,粒度划分的决定性因素并非总的数据量,而是总的行数。因为对数据的存取通常是通过存取索引来实现的,而索引是对应表的行来组织的,即在某一索引中每一行总有一个索引项,索引的大小只与表的总行数有关,而与表的数据量无关。

如商场数据仓库的例子,一个商场可以经营上千种甚至更多的商品。商品的来源也有许多。每日的商品销售数据更是不计其数,每时每刻都在生成新的记录,进入“商品”主题的数据量是很大的,因而最好采用多重粒度,如对商品销售的分析主要是进行销售统计以及销售趋势分析。因此,定义商品销售数据的综合层次要更丰富一些,如每种商品(按商品号)的周统计销售数据、月统计销售数据以及季统计销售数据,每类商品(按商品类型)的周统计销售数据、月统计销售数据以及季统计销售数据,等等。

3. 关系模式定义

数据仓库的数据最终将以关系数据库显示和存储。每个主题都是由多个表来实现的,这些表之间依靠主题的公共码键联系在一起,形成一个完整的主题。在概念模型设计时就

确定了数据仓库的基本主题，并对每个主题的公共码键、基本内容等做了描述。在这一步里将要对选定的当前实施的主题进行模式划分，形成多个表，并确定各个表的关系模式。

如对“商品”主题，考虑粒度划分层次，有如下关系表的内容。

公共码键：商品号。

(1) 商品固有信息：

商品表(商品号、商品名、类型、颜色、价格……)——细节级。

(2) 商品销售信息：

销售表1(商品号、客户号、销售日期、售价、销售量……)——细节级。

销售表2(商品号、时间段1、销售总量……)——综合级。

……

销售表n(商品号、时间段n、销售总量……)——综合级。

4. 定义记录系统

数据仓库中的数据来源于多个已经存在的操作型系统及外部系统。定义记录系统是建立数据仓库中的数据以源系统中的数据的对照记录。由于各个源系统的数据都是面向应用的，不能完整地描述企业中的主题域，并且多个数据源的数据存在着许多不一致。因此要从数据仓库的概念模型出发，结合主题的多个表的关系模式，需要确定现有系统的哪些数据能较好地适应数据仓库的需要。这就要求选择最完整、最及时、最准确、最接近外部实体源的数据作为记录系统，同时这些数据所在的表的关系模式最接近构成主题的多个表的关系模式。记录系统的定义要记入数据仓库的元数据。

以商场的数据仓库为例，“商品”主题的有关内容分散在原有的销售子系统、库存子系统、采购子系统等操作型的数据库中。不同数据源有关商品的信息有相交的部分，可能存在不一致的信息。从记录系统的要求出发，选择原有的分散数据库中最接近外部实体源的数据，定义为数据仓库的记录系统。商品主题的记录系统在元数据中的描述，如表4.2所示。

表4.2 记录系统的定义

主题名	属性名	数据源系统	源表名	源属性名
商品	商品号	库存子系统	商品	商品号
商品	商品名	库存子系统	商品	商品名
商品	类别	库存子系统	商品	类别
商品	客户号	销售子系统	客户	客户号
商品	销售日期	销售子系统	销售	日期
商品	售价	销售子系统	销售	单价
商品	销售量	销售子系统	销售	数量
商品	库存量	库存子系统	库存	库存量
商品	库存号	库存子系统	仓库	仓库号

注：数据仓库中主题中的属性名要统一规范化。各源系统中的数据库中相关属性名，去掉不要的属性项，作为数据仓库和源系统的对比说明(记录系统的定义)放入元数据中。

4.1.4 物理模型设计

数据仓库的物理模型设计是为逻辑模型设计的数据模型确定一个最适合应用要求的物理结构(包括存储结构和存取方法)。

物理模型的设计所做的工作是估计存储容量,确定数据的存储结构,确定索引,确定数据存放位置,确定存储分配。

1. 估计存储容量

物理模型重点在于物理存储,随着数据仓库的增大,需要知道最初和后来需要多少存储空间。

(1) 对每一个数据库表确定数据量

① 行(记录行)数的初始估计;

② 行的平均长度;

③ 估计行的每月增长数;

④ 表的初始大小,以兆字节(MB)计算;

⑤ 表按时间 6 个月和 12 个月存储的数据大小。

(2) 对所有的表确定索引

① 索引的个数;

② 索引对最初、6 个月和 12 个月存储数据所需要的空间。

(3) 估计临时存储

① 排序、合并需要的临时空间;

② 准备区(大量数据交换的场所)内的临时文件;

③ 准备区内的永久文件。

2. 确定数据的存储计划

确定数据的存储计划包括:

(1) 建立聚集(汇总)计划

假设数据仓库用户有 80%的查询需要汇总信息,这样就应该建立汇总表。如果数据仓库只存储最小粒度的数据,每次查询遍历所有的明细记录,然后生成汇总信息,就要用去大量的时间。聚集(汇总)数据表必须包括在物理模型中。应该建立多少汇总表,这要根据查询需求来决定。

(2) 确定数据分区方案

假设有 4 个维度表,平均每个表有 50 行,对于这些维度表中的行,潜在的事实表将有超过 600 万行记录。事实表非常巨大,大表非常难管理。

分区可以将表分解成易于管理的小表。对事实表的分区并不是简单地分解数量。一般采用按垂直分区或水平分区(即按不同维度分区或按时间顺序分区)制定分区准则(如按产品分组)。除事实表分区外,维表也分区。每个表的分区个数是多少,在表分区后,使查询知道到所需的分区内进行。

(3) 建立聚类选项

在数据仓库中,很多的数据访问是基于对大量数据的顺序访问,这可以通过聚类来提高性能。聚类是将相关的数据放在存储介质的相邻物理块上管理。这种安排使相关联的数据能够在一次输入操作中全部取出,提高查询效率。

3. 确定索引策略

在数据仓库中由于数据量很大,需要对数据的存取路径进行仔细设计和选择,建立专用的复杂的索引,以获得最高的存取效率,因为在数据仓库中的数据是不常更新的,也就是说,每个数据存储是稳定的。虽然建立索引有一定的代价,但是一旦建立就几乎不需要再维护索引。

传统的数据库采用 B-Tree 索引,它是一个高效的索引,如图 4.7 所示。B 树是一个平衡(balance)树,即每个叶结点到根结点的路径长度相同。B 树索引是一个多级索引。每个非叶结点包括多个按顺序排列的关键字值:

$$K_1 < K_2 < \cdots < K_{n-1}$$

每个关键字有一个对应的指针 $P_i(i=1,2,\cdots,n-1)$指向下层结点的指针桶(多个关键字和对应的指针)最小关键字值。叶结点的关键字值的指针指向一个文件记录。

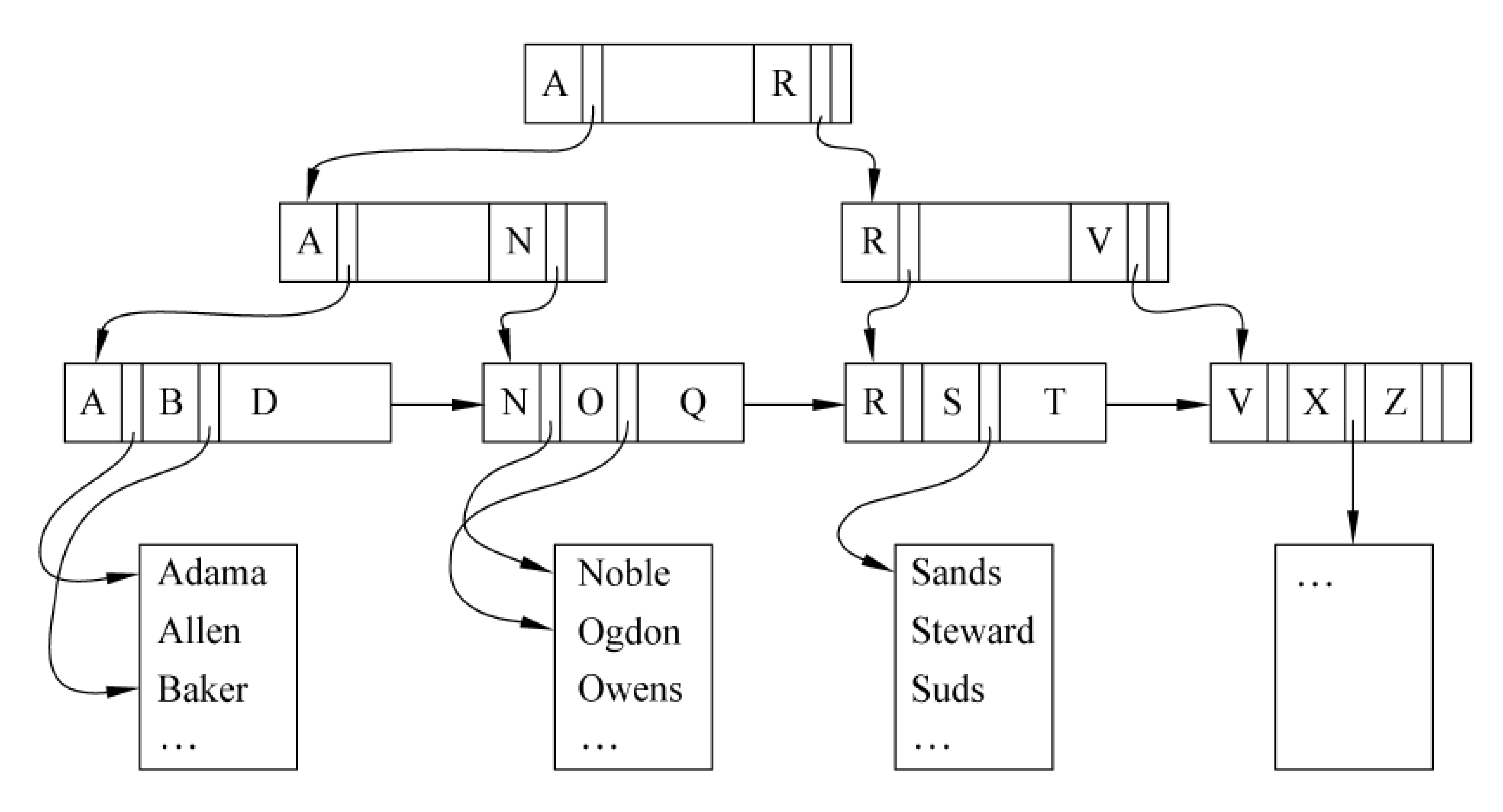


图 4.7 传统 B-Tree 索引

4. 确定数据存放位置

数据仓库中,同一个主题的数据并不要求存放在相同的介质上。在物理设计时,常常要按数据的重要程度、使用频率以及对响应时间的要求进行分类,并将不同类的数据分别存储在不同的存储设备中。重要程度高、经常存取并对响应时间要求高的数据就存放在高速存储设备上,如硬盘;存取频率低或对存取响应时间要求低的数据则可以放在低速存储设备上,如磁盘或磁带。

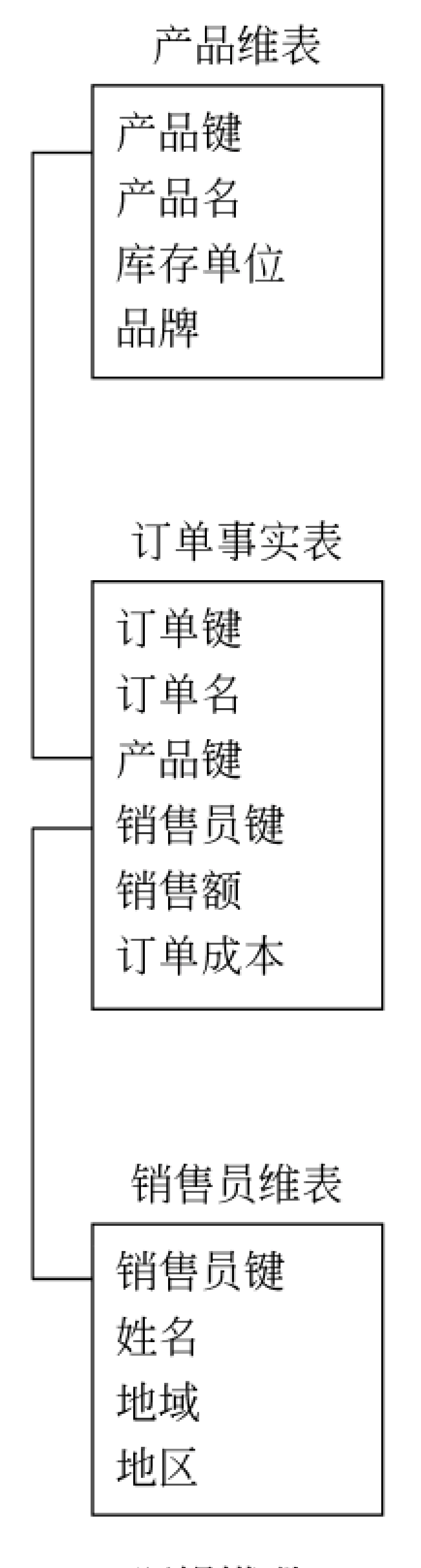
数据存放位置的确定还要考虑一些其他方法,如:决定是否进行合并表;是否对一些经常性的应用建立数据序列;对常用的、不常修改的表或属性是否允许冗余存储。如果采用了这些技术,就要记入元数据。

5. 确定存储分配

物理存储中以文件、块和记录来实现。一个文件包括很多块,每个块包括若干条记录。文件中的块是数据库的数据和内存之间 I/O 传输的基本单位,在那里对数据进行操作。

增大文件中的块大小,即将更多的记录和行可以放入一个块中,因为一次读操作可以读入更多的记录,大块减少了读操作的次数。但是,大块结构对读取记录少时,操作系统也将读入很多不必要的信息到内存中,影响了内存管理。

用一个简例来说明逻辑模型和物理模型的内容,见图 4.8 所示。



(a) 逻辑模型

名称	类型	长度	注释
产品维表			包括公司所有产品的信息
Product-Key	integer	10	主键
Product-Name	char	25	产品名称
Product-SKu	char	20	库存单位
销售员维表			包括不同地区的所有销售员信息
Salpers-Key	integer	15	主键
Salpers-Name	char	30	销售员姓名
Territory	char	20	销售员所在区域
Region	char	20	所在地区
订单事实表			包括公司收到的所有订单
Order-Key	integer	10	订单键
Order-Name	char	20	订单名称
Product-ref	integer	10	参考产品主键
Salpers-ref	integer	15	参考销售员主键
Order-Amount	Num	8，2	销售额
Order-Cost	Num	8，2	订单成本

(b) 物理模型

图 4.8 逻辑模型与物理模型

4.1.5 数据仓库的索引技术

索引技术在于提高数据仓库访问效率。下面介绍 3 种重要的数据仓库索引技术:位索引技术、标识技术与广义索引。

1. 位索引技术

Sybase 公司推出的数据仓库 Sybase IQ,采用位索引(Bit-Wise)技术,它在处理复杂的查询时,比传统数据库索引 B-Tree 有了突破。

(1) Bit-Wise 索引技术

Bit-Wise 索引技术在存储数据的方式上与传统的关系数据库有所不同，它不是以“行记录”而是按“列”为单位存储数据，即对数据进行垂直分割。对于每一个记录的字段满足查询条件的真假值用 1 或 0 的方式表示，或者用该字段中不同取值(即多位二进制)来表示。

一般 DSS 查询往往仅涉及大量数据记录中的少数列，因而不需要访问原始数据就能快速获得查询结果。显然，利用字段的不同取值也能快速进行数据聚类、分组、求最大值、求最小值及求平均值等。

对于高度可选择的数(称高基数)，如姓名或地址等可能有数万个选择值，用(1，0)真假值来索引是不合适的。

例如，检索“美国加州有多少男性未申请保险?”

在数据库中，每个记录中对于性别是男性的字段取值为 1，女性为 0，是加州的字段取值为 1，其他为 0，对于未参加保险的字段取值为 1，参加的取值为 0。该 3 列字段值为 1 或 0。对三字段均满足条件记录进行累加。对下面简单数据库利用 Bit-Wise 技术得到两个记录满足条件，如图 4.9 所示。



项目	性别	保险	州
1	M	Y	MA
2	M	N	CA
3	F	Y	IL
4	M	N	CA

男	未保险	加州	
1	0	0	
1	1	1	=2
0	0	0	
1	1	1	

图 4.9　Bit-Wise 索引

(2) B-Tree 技术与 Bit-Wise 索引技术对比

Bit-Wise 索引技术比 B-Tree 技术能提高相应速度 10～100 倍。

① B-Tree 索引技术特点

- 按行存储数据；
- 针对具体查询建立查询驱动的索引机制；
- 存储被索引的字段数据；
- 一列允许一个索引；
- 适合高基数字段。

② Bit-Wise 索引技术特点

- 按列存储数据；
- 针对实际特征建索引；
- 不存储实际索引字段内容；
- 一列允许多个索引；
- 数据压缩技术和位操作技术；
- 适合低基数字段，兼顾高基数字段。

③ 实例比较

对于检索“美国加州有多少男性未申请保险?”为例，假设数据库有 10M 个记录，每个记录长 800B，每一页 16KB。

- 按传统的关系数据库的检索：需要经过 50 万次 I/O 操作。

- 按 Bit-Wise 检索：对于 10M 个记录建立 3 列的 Bit-Wise 索引，共占(10Mb * 3/8)B 的空间，每页 16KB，则这些索引仅占 235 页。存取这些索引只要进行 235 次 I/O 操作即可。

④ B-Tree 不适合数据仓库

- B-Tree 只适合高基数(cardinality)字段：对于高基数字段，如物资编号、顾客编号等具有惟一的数据值，B-Tree 很适合。但对于低基数字段就毫无价值，如性别字段只有男、女两个值，建立 B-Tree 索引就没有意义。
- B-Tree 索引增加了在数据仓库中构造和维护索引的代价：由于 B-Tree 索引包含实际数据和其他信息(如指针等)，因而使得索引需占用一定的空间和时间。如果构造所有相关的索引，数据仓库就会占 2～4 倍原始数据空间。当成批插入删除时，索引就非常敏感，有可能失去平衡并降低性能。通常来说，10%～15%的数据修改会导致重建索引。
- B-Tree 索引不适合复杂查询：B-Tree 用于简单查询及已知公共存取路径的环境下才有优点，而在数据仓库应用中，通常是复杂的查询，并经常带有分组及聚合条件。此时，B-Tree 索引往往是无能为力的。

2. 标识技术

使用标准的数据库技术来储存数据仓库是非常昂贵的。较好的替代方法是用基于标识的技术来储存数据仓库。这种技术根本不同于关系数据库技术。利用关系数据库技术，当加入一个记录到系统中时，会追加此数据的一个物理代表块到磁盘上。假设一些标准数据库管理系统中的样本记录如表 4.3 所示。

表 4.3 样本记录

	姓名	籍贯	职称	年龄		姓名	籍贯	职称	年龄
记录 1	陈文东	江西	教授	56	记录 6	赵玉	吉林	讲师	32
记录 2	何玉辉	河北	讲师	32	记录 7	黄小斌	江苏	讲师	28
记录 3	李宝	湖南	副教授	37	记录 8	赛英花	山东	副教授	32
记录 4	施东	江苏	讲师	28	记录 9	彭宏	江西	讲师	25
记录 5	曹文杰	湖南	副教授	36	记录 10	廖宇宙	湖南	教授	42

每次完成一个事务时，就会添加一个新记录到标准的数据库中。数据的缩放比例是线性的，因为数据量是存放记录的一个函数。但是在如表 4.3 所示的小型的、简单的数据库查看数据记录时，会发现在整个数据库中有数据冗余。例如籍贯“湖南”出现了 3 次，年龄“32”则出现了 3 次，职称“讲师”出现了 5 次。因此这个数据库中有明显的物理冗余。

假设可以为此数据库中的每个实体创建一个标识。“江西”在籍贯中是 01 标识。“28”在年龄中是 02 标识。“讲师”在职称名中有一个 03 标识。上面的数据库可以被简化为一系列标识，如表 4.4 所示。

表　4.4

姓名	籍贯	职称	年龄	姓名	籍贯	职称	年龄
陈文东 01	江西 01	教授　01	25 01	赵玉　06	山东 06		42 06
何玉辉 02	河北 02	副教授 02	28 02	黄小斌 07			56 07
李宝　03	湖南 03	讲师　03	32 03	赛英花 08			
施东　04	江苏 04		36 04	彭宏　09			
曹文杰 05	吉林 05		37 05	廖宇宙 10			

一旦建立完这些标识，数据库可被精简，如表 4.5 所示。

表　4.5

记录 1 01,01,01,07	记录 6 06,05,03,03
记录 2 02,02,03,03	记录 7 07,04,03,02
记录 3 03,03,02,05	记录 8 08,06,02,03
记录 4 04,04,03,02	记录 9 09,01,03,01
记录 5 05,03,02,05	记录 10 10,03,01,06

记录被标识以后，存储这些记录的空间将大大缩小。此外，数据量越大(也就是记录量越多)，标准的数据库和标识数据库的存储需求差异也就越大。换句话说，记录量越多，基于标识的数据库的优势就越明显。使用标识数据库技术时，有几项是非常有利的应用：

- 大量压缩数据。
- 数据越多，标识数据比标准的、基于记录的数据更有利。
- 因为数据被大量压缩，所以整个数据库可以存放在内存中。
- 可以索引所有的行和所有的列。

一旦将基于标识的数据库存放在内存中，处理速度会得到很大的提高。根据不同的细节，查询的速度可以提高 2 到 3 个(甚至更多)数量级。提高了处理速度，很多工作就会成为现实，例如，分析员可以很容易地进行扫描整个数据库的查询。

大量压缩数据的另一个主要益处就是索引所有属性成为可能。一旦可以索引所有属性，对数据仓库的探索分析就没有限制。分析员可以用任何需要的方式查看任意字段。查询的速度就像这样：如果分析员要精练结果，可以重新书写一个查询公式并重新运行。所有的这些重写公式表示和重新计算都可以在很少的时间里完成，这个时间远远少于标准的基于记录的数据库所需要的时间。事实上，探索数据仓库的功效依赖于基于标识的数据库技术。

3. 广义索引

对数据仓库的一个很广泛的应用问题是“这个月销售最好和最差的 10 种商品是哪些?”可以设计这么一块“黑板”，在上面标明当月销售最好和最差的 10 种商品的名称或者它们相关记录的存放地址。这块“黑板”就是人们所说的“广义索引”。

数据仓库的数据量巨大，所以要依靠各种各样的索引技术来提高涉及大数据量的查询速度。“广义索引”对于处理如上的最值(最大值或最小值)问题时，其效果是非常明显的，也

较易于实现。在从操作型环境抽取数据并向数据仓库中装载的同时，就可以根据用户的需要建立许多这样的“广义索引”。每次数据仓库装载时，就重新生成这些“广义索引”的内容。这样并不需要为了建立“广义索引”而去扫描数据仓库。而且这些索引都非常小，开销也是相当小，但它给应用所带来的便利却是显而易见的。对于一些经常性的查询，利用一个规模小得多的“广义索引”总比去搜索一个大得多的关系表方便得多。

但是，同时出现的问题就是，随着数据仓库“年龄”的增长以及数据仓库随时间变化的特性，这种“广义索引”的数目也就会成倍的增长，管理这些数目多、规模小、名目繁多的“广义索引”也就成为一件非常棘手的事情。这就需要在元数据中完整地定义说明这些“广义索引”。应用需要时，首先去查找元数据，再去查找相应的“广义索引”或表。

4.2 数据仓库开发

4.2.1 数据仓库开发过程

数据仓库的开发主要是围绕数据仓库功能展开的。数据仓库的主要功能包括数据获取、数据存储和决策分析，这 3 个功能模块组成了数据仓库的体系结构。数据仓库随着决策需求的扩大，数据仓库的数据将迅速增长。这样，数据仓库的开发要适应这种变化，采用螺旋式周期性的开发方法比较合适。

数据仓库的开发过程分为 4 个阶段 12 个具体步骤，如图 4.10 所示。

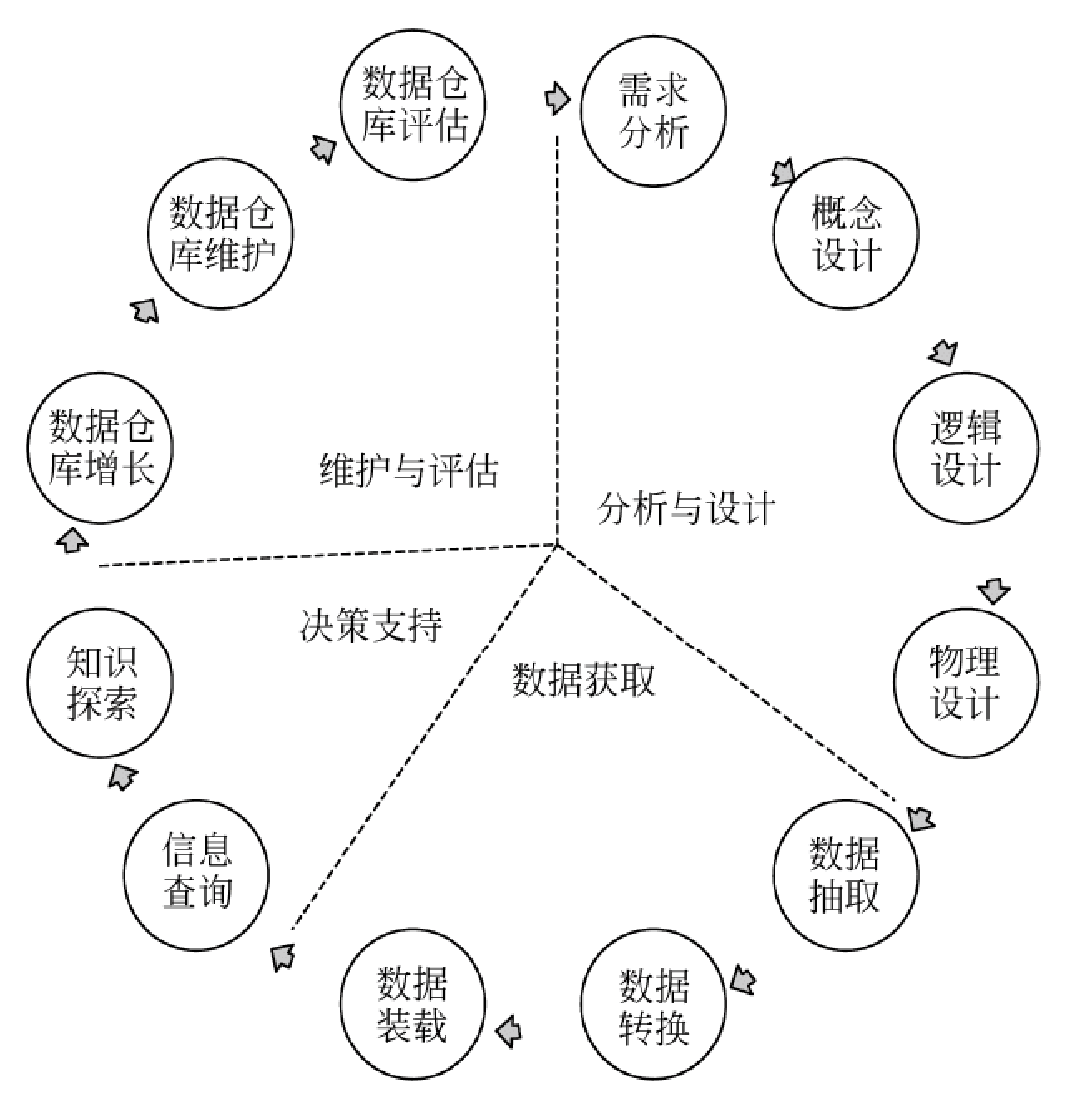


图 4.10 数据仓库开发过程

4.2.1.1 分析与设计阶段

数据仓库开发需要明确如下问题：

(1) 数据仓库开发的范围多大？包括数据的范围、技术的作用(要用到新技术吗?)以及时间上的考虑(开发工作需要在多长时间完成?)。

(2) 企业业务方面的驱动因素是什么，要解决的业务问题是什么？

(3) 开发的数据仓库的决策支持能力是什么？

数据仓库开发的分析和设计阶段包括需求分析、概念设计、逻辑设计和物理设计 4 个步骤。

1. 需求分析

数据仓库的需求分析是根据用户的决策支持需求，确定决策主题域，并分析主题域的商业维度，同时分析支持决策的数据来源，以及向决策主题数据的转换；整个数据仓库的数据量大小以及数据更新的频率确定决策分析方法等。

需求分析是设计和实现数据仓库的基础。例如，银行业数据仓库的需求分析包括以下几点。

(1) 决策支持需求：在竞争性的市场中银行决策者认识到，必须利用其日常活动中包含的大量信息，预测信用卡使用状况和利润率的能力。

(2) 信息需求：对最终用户进行调查，以确定哪些信息有助于销售或有助于调整银行的信息政策。

(3) 业务需求：定义销售信息处理、信息的类型和销售渠道。

(4) 用户访问需求：确定用户访问数据仓库所需的时间，以及数据访问的偏好。

(5) 选择主题：选择一个主题区——“信用卡”。

(6) 初始规模：确定主题域的数据量。

2. 概念设计

在数据仓库中的概念模型设计中，需要确定主题域及其内容。利用需求分析的结果建立概念模型，即对每个决策主题与属性以及主题之间的关系用 E-R 图模型表示出来。E-R 图能有效地将现实世界表示成信息世界，也便利向计算机的表示形式进行转化。

例如，银行业信用卡主题域分析如下。

(1) 主题域范围：确定了“信用卡”主题域，对某些实体，如顾客，要求在这一主题域发挥作用。

(2) 所需细节水平：为支持概括和趋势计算，需要存入持卡人的日常活动。

(3) 初步概括表：对“信用卡”主题需要建立初步概括表，按行业和地理特征进行概括，对概括时段确定为每月。

3. 逻辑设计

在逻辑模型设计中，需要分析主题域，将概念模型(E-R 图)转换成逻辑模型，即计算机

表示的数据模型。数据仓库的数据模型一般采用星型模型。

逻辑设计中还需要进行数据粒度层次的划分，星型模型中事实表、维表的关系模式定义，数据转换的记录系统的定义。

银行业信用卡主题的逻辑模型是多维表的星型模型，需要将概念模型的 E-R 图转换成星型模型。

4. 物理设计

数据仓库的物理模型设计是对逻辑模型设计的数据模型确定物理存储结构和存取方法。数据仓库的星型模型在计算机中仍用关系型数据库存储。

物理设计还需要进行存储容量的估计；确定数据存储的计划；确定索引策略；确定数据存放位置以及确定存储分配。

例如，银行业的物理数据库设计包括以下内容。

(1) 数据库设计：对主题中的事实表和维表设计数据库存储结构和存放位置。

(2) 概括表：按行业代码或按月建立一个概括表。

(3) 索引：对数据仓库中的数据建立多种索引。

(4) 建立备份和恢复准则：使数据仓库能适应不同的备份和恢复。为了防止数据损失，需要对文件进行备份。

4.2.1.2 数据获取阶段

数据获取阶段包括数据抽取、数据转换、数据装载 3 个步骤。

数据仓库中的数据主要来源于事务处理(操作型)系统中的数据。由于数据仓库对数据的使用目的与事务处理对数据的使用目的不同，这就形成了对事务处理系统中的数据的抽取，并进行转换，按数据仓库的数据存储要求装载数据。

1. 数据抽取

数据抽取工作主要进行数据源的确认，确定数据抽取技术，确认数据抽取频率，按照时间要求抽取数据。

由于源系统的差异性，如计算机平台、操作系统、数据库管理系统、网络协议等的不同造成了抽取数据的困难。

2. 数据转换

数据抽取得到的数据是不能直接存入数据仓库的。数据转换工作包括：数据格式的修正、字段的解码、单个字段的分离、信息的合并、变量单位的转化、时间的转化、数据汇总等。

3. 数据装载

经过数据转换的数据装入数据仓库有 3 种类型：

- 初始装载：第一次装入数据仓库。
- 增量装载：根据定期应用需求装入数据仓库。

• 完全刷新：完全删除现有数据，重新装入新的数据。

数据装载时，一般利用选定的批量装载程序，目的是高效和及时地把数据装载到数据仓库中去。

例如，银行业的数据仓库的数据获取阶段包括：

(1) 候选数据源：给定数据需求和粒度需求，指定日常事务文件为关键数据源。

(2) 完整性：检查数据来源的完整性。

(3) 评价：对数据源进行评价。

(4) 数据转换：将数据源中的数据变换到目的地去，同时保持数据准确性和完整性的过程。

(5) 数据装载：将数据转换后的数据加载到目的文件和平台上去。可以用查询来验证业务报表的内容。

(6) 评审过程：开发评审程序来验证是否所有的信用卡事务都发生在指定的时间期限内。

(7) 元数据的加载：除加载一般的元数据外，还要加载有特别用途的元数据，如在特殊环境中，反映数据变化的元数据。

(8) 系统测试：系统测试用以保证各部分能相互配合，并维护数据的完整性。

4.2.1.3 决策支持阶段

数据仓库的建立就是要达到决策支持的目的。决策支持阶段包括信息查询和知识探索两个步骤。

数据仓库有两类用户，一类是信息查询者，是数据仓库的主要用户，用一种可预测的、重复性的方式使用数据仓库，以达到常规决策支持要求。另一类是知识探索者，是数据仓库的少量用户，用一种完全不可预测的非重复性的方式使用数据仓库，以达到挖掘未知知识的要求，取得更大决策支持的效果。这两类不同的用户使用数据仓库，需要具有不同的性能或工具来满足要求。

1. 信息查询

信息查询者使用数据仓库能发现目前存在的问题，例如，发现公司正在流失客户。

为适应信息查询者的要求，数据仓库一般采用如下的方法提高信息查询效率：

(1) 创建数据陈列

对一些分散存放的不同物理位置的数据(如不同月份的数据)，创建一个数据陈列，将相关的数据(每月的数据)放在同一个物理位置上。这样可以提高可预测的和有规律的数据查询效果。

(2) 预连接表格

对于两个或多个表格共享一个公用链或者共同使用的表格，可以将多个表格合并在一个物理表格中，提高数据的访问效率。

(3) 预聚集数据

利用“滚动概括”结构来组织数据。当数据输入到数据仓库时，以每天为基础存储数据。

在一周结束时，以每周为基础存储数据(即累加每天的数据)。月末时，则以每月为基础存储数据。通过这种方式来组织数据，可以极大地减少存储数据所需要的空间并潜在地提高性能。

(4) 聚类数据

聚类将数据放置在同一地点，这样可以提高对聚类数据的查询。

2. 知识探索

知识探索者使用数据仓库能对发现的问题找出原因。例如，找出流失客户的原因。

知识探索者通常用随意的、非重复的方式来查看大量的数据。为满足探索者对大量数据的需要，一般创建一个单独的探索仓库。这样，既不影响数据仓库的常规用户，又可以采用"标识技术"把数据压缩到能将数据放置在内存中，提高数据分析速度。

知识探索者一般使用一些模型帮助决策分析，例如客户分段、欺诈监测、信用风险、客户生存期、渠道响应、推销响应等模型。通过模型的计算来得出一些有价值的商业知识。

知识探索者大量采用数据挖掘工具来获取商业知识。例如，通过数据挖掘得到如下一些知识：

- 哪些商品一起销售好?
- 哪些商业事务处理可能带有欺诈性?
- 高价值客户的共同点是什么?

知识探索者获取的知识为企业领导者提供决策支持，达到保留客户、减少欺诈、提高公司利润等具有重要作用。

4.2.1.4 维护与评估阶段

该阶段包括数据仓库增长、数据仓库维护、数据仓库评估 3 个步骤。

1. 数据仓库增长

数据仓库建立以后，随着用户的不断增加，时间的增长，用户查询需求更多，数据会迅速增长。造成这种增长的原因有：详细数据和汇总数据的增加，历史数据的增加；满足更多用户决策需求、数据的增加等。数据仓库在使用后不断增长已成为数据仓库的特点。

在数据仓库的开发过程中需要适应数据仓库不断增长的现实。

2. 数据仓库维护

数据仓库维护包括适应数据仓库增长的维护和正常系统维护两类。

适应数据仓库增长的维护包括：数据增长的处理、存储空间的处理、ETL 处理、数据模型的修订、增强决策支持的处理等。其中，数据增长的处理工作有：去掉没有用的历史数据；根据用户使用的情况，取消某些细节数据和无用的汇总数据，增加些实用的汇总数据。

存储空间的处理工作主要是对增长的存储设备要有计划。存储成本是软件成本的 4～5 倍。

正常的系统维护工作包括数据仓库的备份和恢复。由于数据仓库的数据是经过了复杂

的清洗和转换过程得到的，它代表企业的丰富历史，能适应用户信息查询和决策支持。备份数据内容是非常必要的。备份数据也为系统恢复提供基础，一旦系统出现灾难，利用备份数据可以很快地将数据仓库恢复到正常状态。

3. 数据仓库评估

数据仓库评估包括 3 个方面：系统性能评定；投资回报分析；数据质量评估。

（1）系统性能评定

系统性能评定包括：

- 硬件平台是否能够支持大数据量的工作和多类用户、多种工具的大量需求？
- 软件平台是否适用一个高效的且优化的方式来组织和管理数据？
- 是否适应系统（数据和处理）的扩展？

（2）投资回报分析

投资回报分析包括定量分析和定性分析。

- 定量分析是计算投资回报率（ROI），即收益与成本的比率。按 IDC（加拿大）公司提供的数据表明：欧美 62 家企业建立的数据仓库 3 年投资回报率平均值为 401%，收回投资的平均时间为 2.3 年。最终用户获得的效益大约占总效益的 50%，信息收集人员和维护人员获得的效益共占总效益的 50%。

IDC 的调查结果表明，对于环境比较复杂的企业，数据仓库是一种有价值的投资。

- 定性分析是分析如下几个方面的效果：企业与客户之间的关系状态如何？给客户获得的好处如何？建立企业的合作关系如何？对转瞬即逝的机会快速反应能力如何？管理宏观和微观数据的能力如何？改善管理能力如何？

（3）数据质量评估

数据质量是数据仓库成功的关键，只有高质量的数据才能为决策支持提供准确的依据。保证决策的正确性。

数据质量的评估标准有：

- 数据是准确的。数据必须保证它的准确性，如姓名、地址对营销部门必须正确。
- 数据符合它的类型要求和取值要求。定义了数据字段类型（如字符型、实数型等）后，对该字段的所有数据必须满足类型要求，其取值必须在指定的范围内。如“性别”字段是“字符型”，其取值范围只有“男”或“女”。
- 数据具有完整性和一致性。数据的完整性体现在对不同的需求，都应该获得所需要的数值，不应该有缺失值。数据的一致性体现在相同记录下同一字段的数据在多个不同的源系统中有相同的类型和取值。如产品 ABC 的代码是 1234，在不同的源系统中都应该是一致的。
- 数据是清晰的且符合商业规则。数据正确的命名可以帮助用户更好地理解数据元素，如果用户不了解它的含义就不可能很好地使用它。数据必须符合商业规则，如销售价格不能低于底价，贷款余额不能是负值。
- 数据保持时效性并不能出现异常。对不同时间要求的数据（如按照月）能按时提供，保持时效性。数据不能出现异常，如客户的通信地址不能是传真号码或者电话

号码。

4.2.2 数据质量与数据清洗

数据质量是数据仓库的成功关键。完整而准确的数据能够大大提高客户服务的质量，给产品提高交叉销售的机会(即购买一个产品时，可能购买其他产品)，高质量的数据能减少成本和风险，提高生产率，完成实时的信息分析，最本质的是保证战略决策的制定。

在数据仓库的开发中，数据的抽取和转换过程中会发现数据质量问题，要及时找出数据污染的原因，进行有效的数据清洗，确保数据的高质量。

1. 数据质量问题

数据质量问题表现为：

(1) 字段中的虚假值。在输入数据时，有时会将字母 P、O 等，误改成数字 9 和 0。

(2) 数据值缺失。这在客户数据中经常出现。

(3) 不一致的值。不同的源系统代码表示不一致。如有的代码表示为 A(Auto)、H(Home)、F(Flood)；有的表示为 1、2、3；有的表示为 AU、HO、FL 等。

(4) 违反常规的不正确值。如一年工作的天数，加上假日、病假天数，超过 365 天。

(5) 一个字段有多种用途。一个字段同一数据在不同部门可能有不同的含义。

(6) 标法不惟一。例如销售系统与库存系统的产品代码不一致。

2. 数据污染产生的原因

数据被污染所产生的原因有：

(1) 系统转换。由于系统升级而发生变化时，在文件转换过程中，会对数据产生污染。系统转换和迁移是数据污染的重要原因。查找数据污染需要了解每一次源系统所经过的转换过程。

(2) 数据老化。在源系统中有很多旧系统时，旧的值随着时间的变化会失去它的含义和意义，逐渐形成数据污染。

(3) 复杂的系统集成。数据不一致会产生数据污染。数据仓库的源系统种类越多，出现污染数据的可能性越大。

(4) 数据输入的不完整信息。在初始数据输入时，没有完全输入所有的字段，将导致数据值缺失；对必须输入的字段，随便输入一些通用数据，都将产生数据污染。

(5) 输入错误。错误的数据输入也是数据污染的一个主要来源。

(6) 欺诈。有些人为了欺诈，千方百计往系统中输入错误的数据，特别是涉及金额或产品数量的字段。

(7) 缺乏相关政策。当公司对数据质量没有明确的相关政策，它的数据质量就不可能得到保证。

3. 数据清洗

清洗数据仓库中所有数据的成本是相当高的。在现实世界中，绝对的高质量数据是不

现实的，不能期望100%的数据质量。清洗数据采用“面向目标”的原则，先确定要使用哪些数据，然后确定目标是什么。清洗数据要明确如下问题：

(1) 需要清洗哪些数据

清洗哪些数据是根据数据仓库要回答用户的问题类型，找出回答问题所需要的数据。权衡每部分数据的价值，并估计对数据清洗、对用户分析会造成什么影响。通常只清洗那些重要的数据，而忽略那些不重要的数据。

(2) 在什么地方清洗

数据的错误来自源系统，在数据进入数据仓库之后再进行清洗是不现实的，这样会破坏已转移和转载其他数据。通常，数据在被存储进数据仓库之前就应该进行清洗。数据抽取过程中被抽取的数据一般进入缓存区域，数据装载过程从缓存区域进入数据仓库中。

在缓存区域中清洗数据相对容易。

(3) 怎么清洗

清洗源系统中的数据，必须找到适合源系统的字段和格式的清洗工具。现在已有很多完成各种数据清洗功能的工具软件可以采用。对于特殊的数据污染则要专门编制程序来完成数据清洗。

对于要净化的数据元素分为3个优先级类型：高优先级、中优先级和低优先级。对高优先级的数据要达到100%的数据质量等级。中优先级的数据越准确越好，对这类数据，要在数据修正的成本和坏数据可能造成的影响之间进行平衡。低优先级的数据可以在有时间和需要的时候进行清洗。

(4) 建立一个数据质量框架

数据质量框架包括：建立数据质量领导小组；建立数据质量政策和标准；定义质量指标参数和基准；识别受坏数据影响最大的商业功能；选择那些有较大影响力的数据元素，确定优先级；对有较大影响力的数据元素制定清洗计划，并执行数据清洗；再为较小影响的数据元素制定清洗计划，并执行数据清洗。这个框架是确保数据质量的基础。

4.2.3 数据粒度与维度建模

数据粒度是指数据仓库的数据中保存数据的细化程度或综合程度的级别。细化程度越细，粒度级别就越小；相反，细化程度越粗，粒度级别就越高。

数据粒度深深影响存放在数据仓库中的数据量的大小，同时影响数据仓库所能回答的查询类型。

数据仓库的设计需要在数据量大小与查询的详细程度之间作出权衡。

例如，在数据仓库中存储一个顾客（张三）一个月的每个电话的细节，能够查询出“张三在某日给女友是否打过电话”。其存储量是每个月200个记录40 000个字节。若存储一个顾客一个月的电话综合，能够查询“张三这个月打了多少个长途电话”。其存储量是每个月一个记录200个字节。

1. 大维度与雪花模型

大维度表现在两方面：大维度表的记录数很大；大维度表的属性很多。在数据仓库中，

客户维度和产品维度是典型的大维度。一个全国连锁店的客户维度可能包括上亿条记录。大型零售店的产品维度也相当巨大。

一般大型客户维度有 2000 万条记录,150 个维度属性,可能有多种层次结构。一般大型产品维度有 100 000 种产品,100 多个属性,也有多种层次结构。大维度数据仓库运行时会很慢,效率很低。大维度表采用雪花模型的数据组织,是一种有效的方法。

对产品维度而言,产品分属于产品品牌,品牌又分属于产品分类。对客户维度而言,客户分属于地区,地区分属于国家。以上结构采用雪花模型的数据组织,将减少各维表的记录数,使查询过程中搜索记录数目减少。对于销售的雪花模型,如图 4.11 所示。

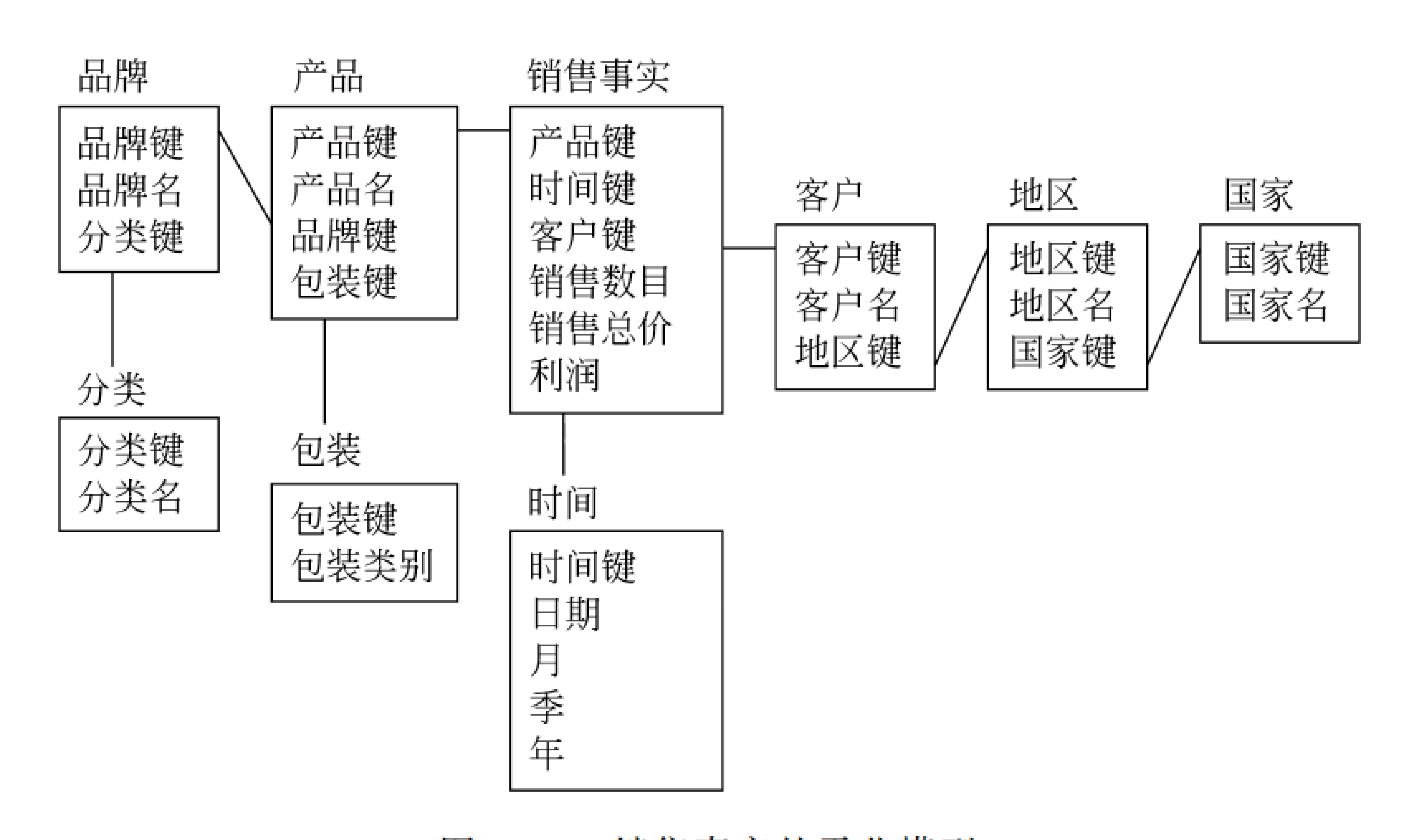


图 4.11 销售事实的雪花模型

2. 综合事实表

在基础事实表中,各条记录反映维度多层结构中最低层次的数据。例如,销售事实是某日、某个商店和某个产品相关的销售数量和销售总价。

在现实中,大多数查询不是基于基础事实表上操作的,而是基于综合数据的查询。这样建立综合事实表是提高综合数据查询的非常有效的方法,且大大提高数据仓库的性能。

在多维表中,很多维都具有层次结构,对不同维的层次的提升,可建立多种综合事实表。综合事实表是由基础事实表衍生出来的。同时维度也将衍生出高层次的维表,它与综合事实表连接起来一起使用。

例如,对产品维从每一个具体的产品上升为分类产品,建立分类产品维表。按照分类产品键来综合基础销售事实表的事实,形成(衍生)综合销售事实表,如图 4.12 所示。

从图 4.12 可见,对基础事实表查询,利用产品维表;对综合事实表查询,利用产品分类维表。

以上是对一个维度进行提升产生的综合事实表和衍生维表。若对 2 个或 3 个维度同时进行提升,所产生的综合事实表也需要衍生出相应 2 个或 3 个高层次的维表。综合事实表将大大提高综合数据的查询效果。

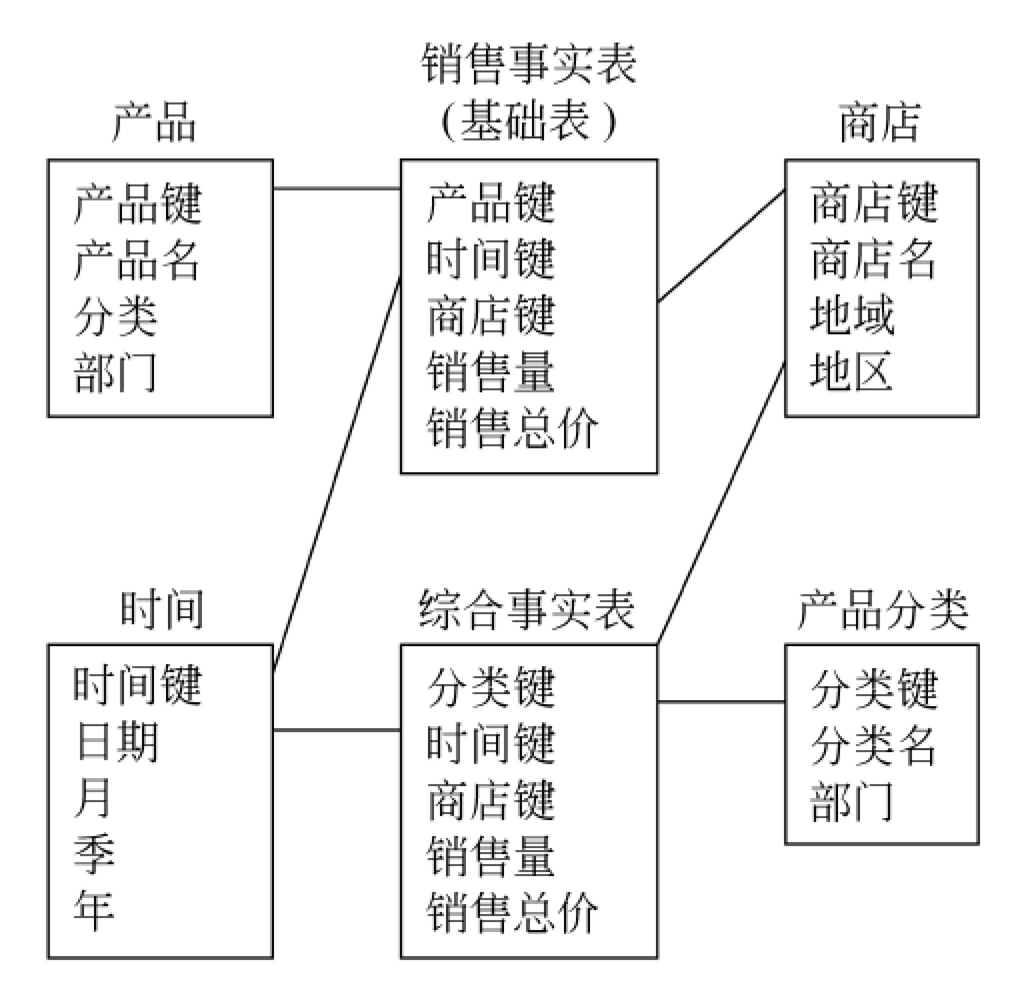


图 4.12　综合事实表和衍生维度(产品分类)表

4.3　数据仓库技术与开发的困难

4.3.1　数据仓库技术

数据仓库环境中的数据处理可以概括为装入与访问两个过程。数据从大量数据库中的集成、转换和装载到数据仓库中去。数据一旦被装入,通常是不更新的。数据到数据仓库后将被访问和分析。

1. 管理大量数据

对于数据仓库最重要的技术就是能够管理大量的数据。传统数据库环境和数据仓库环境的一个重要的区别在于,数据仓库中有更多的数据量,比一般的数据库环境中要多得多。数据仓库中的数据量是10GB或100GB级的,而一个通用的DBMS通常管理的数据是MB级的。数据仓库要管理大量的数据,是因为它们包括:

(1) 粒状的、原子的细节;

(2) 历史数据;

(3) 细节和汇总数据;

(4) 元数据。

有好多种管理大量数据的方法:寻址、索引、数据的外延、有效的溢出管理等。管理大量的数据有两方面:能够管理大量数据的能力和能够高效管理数据的能力。任何声称支持数据仓库的技术一定都要满足能力与效率的要求。数据仓库开发者建造数据仓库时,要能够满足处理大量数据的需求。

2. 数据的高效装入和数据压缩

(1) 装入数据

数据仓库的一个重要技术就是能够高效地装入数据。

有好多种装入数据的方法：通过一个语言接口一次一条记录或者一起使用一个程序一次全都装入。另外，在装入数据的同时，索引也要高效地装入。有时，为了平衡工作负载，数据索引的装入可以推迟。

如果数据仓库中数据的装入有不可克服的困难，那么这个数据仓库就没有用处了。

(2) 数据压缩

数据仓库的成功之处就在于能够管理大量的数据。达到这一目的的中心环节是数据的压缩。当数据能够被压缩时，便能存储在很小的空间中。这与数据仓库的环境有关，因为数据在插入到数据仓库中后，是很少被更新的。数据仓库中数据的稳定性减少了空间管理问题，这些问题是在更新紧密压缩的数据时发生的。

压缩的另一个好处是程序员可以完全脱离给定的输入输出操作。当然，对数据的访问就会有相应的解压缩的问题。虽然解压缩需要一定的开销，但这个开销不是 I/O 资源的开销，而是 CPU 的开销。通常，在数据仓库环境中 I/O 资源比 CPU 资源少得多，因此数据的解压缩并不是一个主要的问题。

3. 存储介质的管理

在处理大量数据时，为了满足高效率和合理的费用，应用在数据仓库中的基本技术应该能够解决多种存储介质的问题。仅仅在直接存取存储设备(如磁盘)上管理一个成熟的数据仓库是不够的。考虑到访问速度和存储费用，对数据的存储要分层次，层次的区分如表 4.6 所示。

表 4.6 对数据的存储层次

存储介质	访问速度	存储费用
主存	非常快	非常贵
扩展内存	非常快	贵
高速缓存	非常快	贵
磁盘	快	适中
光盘	不慢	不贵
微缩胶片	慢	便宜

由于数据仓库中的大量数量和被访问到的可能性这两方面的因素存在，一个满载的数据仓库应该放在多种存储层次上。处理数据仓库技术要能管理多种存储介质上的数据。

4. 元数据管理

数据仓库中的元数据比在传统的数据库中更重要。为了更加有效，数据仓库的用户应该能够对准确和实时的元数据进行访问。没有一个好的元数据来源进行运作的话，DSS 分析员的工作就非常困难。典型的元数据包括：

(1) 数据仓库表的结构；

(2) 数据仓库表的属性；

(3) 数据仓库的源数据；

(4) 源数据到数据仓库的映射;
(5) 数据模型的规格说明;
(6) 抽取日志;
(7) 访问数据的公用例行程序。

5. 数据仓库语言

数据仓库需要有非常丰富的数据仓库语言。这种语言是用来有效管理数据仓库中数据和快速、高效访问数据仓库中的数据。

典型的数据仓库语言:
(1) 能够一次访问一组数据;
(2) 能够一次访问一条记录;
(3) 特别要保证,为了满足某个访问要求能够支持一个或多个索引;
(4) 有 SQL 接口;
(5) 能够插入、删除、更新数据。

6. 高效索引

数据仓库的灵魂在于灵活性和对数据的不可预测的访问。这一点也是要求能够对数据进行快速和方便的访问。数据仓库中的数据如果不能方便和有效地检索,那么建立数据仓库这项工作就不是成功的。当然,设计者可以利用许多方法来使数据尽可能地灵活,例如利用双重粒度级和数据分割。但这些技术一定要支持方便的索引,建立和应用索引的费用不能太高。

数据仓库技术不仅必须能够方便地支持新索引的创建和装入,而且要能够高效地访问这些索引。有多种方法能够高效地访问索引:
(1) 位索引;
(2) 多级索引;
(3) 将部分或全部索引装入内存;
(4) 当被索引的数据的次序允许压缩时,对索引项进行压缩;
(5) 创建选择索引或范围索引。

7. 数据仓库的特殊管理

(1) 复合键码

数据仓库环境中一种重要的技术是能够支持复合键码。这种键码在数据仓库环境中随处可见,主要是因为数据仓库中数据的随时间变化的特性。

(2) 变长数据

数据仓库环境的另一个重要的技术是有效管理变长数据的能力。

变长数据如果被经常更新和改变,就会产生性能上的严重问题。但当变长数据很稳定,如在数据仓库中时,就没有固有的性能问题。另一方面,由于数据仓库中数据的多样性,对数据的变长结构的支持是强制性的。

(3) 快速恢复

数据仓库环境的一个简单而又重要的技术特性是，能够从非直接存取存储设备快速地恢复数据仓库表。当可以从二级存储设备上恢复时，就可以节约大量开支。如果没有能从二级存储设备上快速恢复的能力，通常的做法是将直接存储设备的数目增加一倍，然后将增加出的数目作为恢复/复原的存储区。

8. 多维 DBMS 和数据仓库

在数据仓库中经常讨论的技术是多维数据库管理系统(多维 DBMS)。多维数据库管理系统提供了一种信息系统结构，使得对数据的访问非常灵活，可以用多种方法对数据进行切片、分割，动态地考察汇总数据和细节数据的关系。多维 DBMS 不仅提供了灵活性，还可以对终端用户进行管理，这些非常适合 DSS 环境。

数据仓库中的细节数据为多维 DBMS 提供了数据源，多维 DBMS 需要定期地刷新。为此，数据要定期从数据仓库中导入到多维 DBMS 中去。数据仓库和多维 DBMS 的区别如下：

(1) 数据仓库有大量的数据；多维 DBMS 中的数据至少要少一个数量级。

(2) 数据仓库只适合于少量的灵活访问；而多维 DBMS 适合大量的非预知的数据的访问和分析。

(3) 数据仓库内存储了很长时间范围内的数据，如 5 年到 10 年；多维 DBMS 中存储着比较短时间范围内的数据。

(4) 数据仓库允许分析人员以受限的形式访问数据，而多维 DBMS 允许自由的访问。

多维 DBMS 和数据仓库有着互补的关系。数据仓库为非常细节的数据提供了基础，而这在多维 DBMS 中通常是不能看到的。数据仓库能容纳非常详细的数据，这些数据在导入多维 DBMS 时被轻度综合了，导入多维 DBMS 后，数据还会被进一步地汇总。在这种模式下，多维 DBMS 可以包含除了非常细节以外的所有数据。使用多维 DBMS 的分析者可以一种灵活和高效的方式对多维 DBMS 中所有不同层次的数据进行钻取。如果需要的话，分析者还可以向下钻取到数据仓库。通过这种方式将数据仓库和多维 DBMS 结合。DSS 分析者可以得到这两者的好处。DSS 分析者大部分时间里可以在多维 DBMS 中享受其操作高效的优点，同时如果需要的话，还可以向下钻取最低层次的细节数据。

数据仓库和多维 DBMS 的另一个互补的方面是汇总的信息在多维 DBMS 中计算和收集后被存储在数据仓库中。通过以这种方式将数据进行聚合，汇总数据在数据仓库中比在多维 DBMS 能够存储更长的时间。数据仓库和多维 DBMS 还有一个方面是互补的。多维 DBMS 存放中等时间长度的数据，依应用的不同从 12 个月到 15 个月。而数据仓库存放数据的时间跨度要大得多，从 5 年到 10 年。考虑到这一点，数据仓库就成为多维 DBMS 分析者进行研究的源泉。多维 DBMS 分析者乐于知道，如果需要的话，有大量的数据是可用的，但在不需要时确用不着在他们的环境中花费存储所有这些数据的代价。

多维 DBMS 有不同的特色。一些多维 DBMS 建立在关系模型上，而一些多维 DBMS 建立在能优化“切片和切块”数据的基础上，在这里数据可以被认为存储在多维立方体内，后

者的技术基础可以称为“立方体基础”。

两种技术基础都支持多维 DBMS 数据集市，但这两种技术基础之间存在一些差异。

多维 DBMS(OLAP)是一种技术，而数据仓库是一种体系结构的基础。这两者之间存在着互补的和共生的关系。最一般的情况下，数据仓库作为多维 DBMS 的基础——从中选出细节数据的一个子集传到多维 DBMS 中，在那里，数据要么被汇总，要么被聚集。

4.3.2 数据仓库开发的困难

数据仓库由于数据量大(具有 GB 级到 TB 级的数据)，数据包括近期、综合、历史等多个层次，还包括元数据，致使数据的存储和管理复杂。数据仓库的应用包括快速查询、多维分析及数据挖掘等多种类型。这样，数据仓库需要一个具有海量存储的硬件平台和一个能进行并行处理的大型数据库系统。大型数据库厂商 NCR 公司提供的数据仓库硬件平台是具有海量并行处理的 WordMark 系列服务器，数据仓库软件是 Teradata 数据库系统，能处理 GB 级到 TB 级的数据，具有很强的并行处理能力和扩展能力。Oracle、IBM、SAS、Microsoft 等公司也都推出了各自的数据仓库商品，它们为开发数据仓库提供了强有力的工具。这些工具极大地推动了数据仓库的发展。但是，在国外仍存在开发数据仓库的失败。这些失败的案例主要反映在错误的认识观念上，它们构成了开发数据仓库的障碍。

国外总结开发数据仓库的典型错误有：

1. 没有理解数据的价值

没有认识到数据的价值，就不会有效地访问数据和挖掘数据中的信息和知识。数据必须共享，才能充分发挥它的价值，那些垄断数据的做法只可能埋没数据的作用，直接影响数据仓库的开发。数据的一致性是数据共享的基础。数据对于不同的人，由于定义的不一致和时间的不一致，就会造成数据的不一致，这会造成对数据理解的不一致和报表的不一致，从而丧失人们对数据的信任，更谈不上辅助决策。

2. 未能理解数据仓库概念

不了解数据仓库的含义，它所能解决的业务问题和它的用途，必然导致数据仓库开发的失败。数据仓库数据不是将大量现行系统中的数据堆积而成的。数据仓库是将现行管理系统中大量数据按决策主题重新组织，通过集成而形成的。数据仓库包含大量的随时间变化的数据，而不进行实时更新。不像现行管理系统中数据进行实时更新，只保留当前准确的数据。在数据仓库中元数据很重要。元数据能够让用户了解数据仓库中有什么数据，它们是如何组织的，对这些数据如何使用。充分理解数据仓库概念，才能充分发挥数据仓库作用。

3. 尚未清楚地了解用户将如何使用数据仓库之前，便贸然开发数据仓库

一个典型的错误观点是：“只要你建好(数据仓库)了，他们就会用”。这种盲目自信的建造数据仓库，用户未参加界定对数据仓库的需求，必然导致数据仓库的失败。

数据仓库的建造必须有用户代表参加。用户代表懂得数据仓库中需要有哪些数据，以

及如何使用数据仓库来改善他们的决策过程。

4. 对数据仓库规模的估计模糊

数据仓库规模包括数据量的多少、用户数量、常规查询所耗费的资源、并发查询数目、对CPU的要求等。

数据仓库中的数据量多少依赖于数据的主题(如顾客、产品、风险管理、收支等)的划分,以及用户人数。数据太多时,将会使数据存储和加载过程耗资巨大,还会造成数据得不到充分利用或根本无人使用它们。

5. 忽视了数据仓库体系结构和数据仓库开发方法

数据仓库体系结构具有3个层次:数据获取、数据存储和分析工具。这个体系结构是建造数据仓库的图纸。

数据仓库的生命周期(DWLC)不同于系统生命周期(SDLC)。DWLC包括调查、分析当前环境、确定需求、确定体系结构、数据仓库设计、开发、实施、数据管理8个阶段。

数据仓库的设计应该采用数据驱动方法,即以数据为基础(尽可能地利用已有的数据、代码等,而不是从无到有),进行从面向应用到面向分析需求的转变,按决策主题存取数据和分析数据,并逐步提高决策效果。

数据仓库中的数据必须保证它的质量,错误的数据会引起错误的决策。数据的粒度水平如何?即数据应该以细节形式存储,还是以概括形式存储,还是两种形式兼有,这应该根据用户需求来确定。

开发只有克服了以上的错误的认识观念,才能真正发挥数据仓库的作用。

习　题

1. 数据仓库的需求分析的任务是什么?
2. 数据仓库系统需要确定的问题有哪些?
3. 实现决策支持所需要的数据包括哪些内容?
4. 什么是概念模型?它的特点是什么?
5. E-R图如何描述概念模型?
6. 比较数据库的概念模型设计与数据仓库的概念模型设计。
7. 解释图4.1概念模型。
8. 什么是逻辑模型?数据仓库的逻辑模型是什么?
9. 数据仓库的逻辑模型与数据库的逻辑模型有什么不同?
10. 举例说明数据仓库的概念模型到逻辑模型的转换。
11. 在数据仓库中为什么要考虑数据的粒度层次划分?
12. 数据仓库的记录系统包含什么内容?举例说明。
13. 什么是物理模型?数据仓库的物理模型设计包括哪些工作?
14. 为什么数据仓库物理模型设计中要建立汇总计划和确定数据分区方案?

15. 说明图 4.8 中逻辑模型与物理模型的区别。

16. 数据仓库索引技术包括哪些内容?

17. 为什么 B-Tree 索引不适合数据仓库?

18. 数据仓库中采用标识技术有什么好处?

19. 数据仓库的广义索引是什么时候建立的(是在建立数据仓库之后,还是在建立数据仓库同时)? 简单说明为什么?

20. 说明数据仓库开发的 4 个阶段和 12 个步骤。

21. 简要说明数据仓库开发的分析与设计阶段的内容。

22. 简要说明数据仓库开发的数据获取阶段的内容。

23. 简要说明数据仓库开发的决策支持阶段的内容。

24. 简要说明数据仓库开发的维护与评估阶段的内容。

25. 数据质量问题表现在哪些方面?

26. 数据污染产生的原因有哪些?

27. 数据清洗要明确的问题有哪些?

28. 为什么大维度表采用雪花模型?

29. 数据仓库技术包括哪些内容?

30. 国外开发数据仓库的错误有哪些?

第5章　数据仓库管理和应用

5.1　数据仓库管理

一个数据仓库建立后，开始时具有少量的用户，不久就会涌现大量的用户。数据仓库中的数据也会随时间的延伸迅速地增长。

为什么数据会增长？主要原因有：

(1) 数据仓库收集历史数据。数据仓库收集的是5年到10年的数据。

(2) 数据仓库包含满足未知需求的数据收集。数据仓库必须同时满足已知的需求和未知的需求。这样，需要将一些无关和不明显的数据合并到数据仓库中，这种需求导致数据存储量的增长。

(3) 数据仓库既包括了详细数据，也包括了概括数据(汇总数据)。其中汇总数据占数据存储的比例很大。

(4) 数据仓库还包含外部数据(例如人口统计数据、心理学数据等)，这些外部数据可以用来支持多种可预测的数据挖掘任务。

建立和维护一个高性能的数据仓库环境，需要对不断增长的用户和大量的数据进行有效的管理，W. H. Inmon等人概括为：用户使用数据仓库的管理、数据管理、平台管理和服务管理。

5.1.1　用户使用数据仓库的管理

数据仓库的用户有两类：信息使用者和探索者。

信息使用者是使用数据仓库的大量用户。信息使用者以一种可预测的、重发性的方式使用数据仓库平台。他们通常查看概括数据或聚集数据，查看相同的商业维度(如产品、客户、时间)和指标(如收入和成本)随时间的发展趋势。他们天天重复同样的活动，很少使用元数据。他们的工作相对来说属于战术性的。

探索者完全不同于信息使用者，他们有一个完全不可预测的、非重复性的数据使用模式。探索者查看海量的详细数据，而概括数据则会妨碍探索者的数据分析。他们经常查看历史数据，而且查看的历史数据的时间要比信息使用者长得多。探索者的任务是寻找公司数据内隐含的价值并且根据过去事件努力预测未来决策的结果。探索者是典型的数据挖掘者。

1. 信息使用者使用数据仓库的性能优化

信息使用者所提交的查询操作是均匀的且有相当少量的数据，他们需要享有好的查询响应时间。数据仓库管理员采取如下方法来支持信息使用者的性能需求：

(1) 非规格化

数据建模和规范化的作用是产生一种完全没有数据冗余的设计方法。但是,有时在数据仓库设计中引入一些有限的数据冗余来提高数据访问效果。例如,在一些数据表中加入相同的量,这是用增加数据存储来换取数据访问的优化(减少查询时间)。

(2) 创建数据阵列

数据仓库管理员发现用户经常同时使用相关类型的数据时,应创建数据阵列,将这些数据单元存储在一起,提高访问效果。

例如,对于每年所有月份的数据,被分别放置在不同的物理位置上,而用户经常要同时查看1月、2月、3月等月份中的数据,这样会花掉很多搜索时间到不同的物理位置上获取数据。一个好的方法是创建数据阵列,将相关联的数据放在同一物理位置,这样可以提高查询效果。

(3) 预连接表格

节省机器资源的最有效的技巧之一就是,基于一个公用键和共同使用的数据将表格合并在一起。

例如,如果有两个或者更多的表格共享一个公用键,或者使用相近的表格,那么可以将多个表格合并到一个物理表格中。这样做可以很大程度地提高数据访问效率。

(4) 预聚集数据

一种非常有用的方法是根据"滚动概括"结构来组织数据。

当数据被输入到数据仓库中时,以每小时为基础存储数据。在一天结束时,以每天为基础存储累加每小时的数据。在一周结束时,以每周为基础存储累加每天的数据。月末时,则以每月为基础存储累加每周的数据。这样,在累加数据后,就删除被累加的细节数据,通过这种方式来组织数据,数据仓库管理者将极大地减少存储数据所需要的空间。并潜在地提高性能。

当然,管理员也会丧失查看已过时的详细数据的能力,越早获取的数据,保留详细数据越少。但是,许多种类型的数据可接收这种处理,例如,可以非常有效地积累销售、产品、市场数据。

(5) 聚类数据

在预测了用户使用需求以及使用规则后,将不同类型的数据并置在一起,即基于产生共同信息将不同类型的数据记录放置在相同的物理位置,为用户查看这些记录,可以在同一地点找到它们,提高了查询效率。

如果使用的是不可预测和不规则的,那么数据聚类毫无意义。

(6) 压缩数据

压缩将节省资源,因为当系统访问一个物理数据块时,压缩将优化所检索的数据量。利用这种方式,压缩可以使可读取的数据量极大,但同时也需要用户有一定的经验。

要注意的是,在不需要数据更新和预测数据使用模式时才可以使用数据压缩,即压缩不改变任何数据(即一旦写入,不允许重写或者更新)。

(7) 定期净化数据

数据仓库管理员通过定期删除数据仓库中不需要的数据,可以为每个用户提高性能。

没有其他任何一种方法比删除不需要的数据对数据仓库更有利。

(8) 合并查询

如果查询定期发生,那么可以通过把这些查询合并到同一个表格中,从而节省大量资源。查询合并的作用就是把扫描数据仓库表格的次数最小化。

合并查询功能的条件有:

- 当有多个查询询问相同的表格时;
- 所访问的表格是一个大表格;
- 用可预测的、有规律的方式来执行查询;
- 这些查询所执行的连接是以一行接一行的方式;
- 这种查询对执行时间不太敏感。

如果查询不符合这些条件,合并查询功能没有任何优势。

应该如何处理合并查询功能?数据仓库管理员收集所有查询需求并合并到一个大型池中。这些查询的焦点是某一个表格-主表格。只要与二级表格的连接经过某一行的连接点,这些查询就可以查看其他表格并作为查询处理的一部分。

从主表格开始,访问每一行。如果某一行符合任意一个查询的任何选择标准,则保留此行,以被分析,否则继续执行下一行。一旦某行被证明令人感兴趣,则将所需要的数据写入到一个工作文件中。如果有多个查询都需要相同的数据(通过连接点),那么这个结果集将被多个查询所标记。

还有不少提高数据仓库性能的方法,可参考有关书籍。

2. 探索者使用数据仓库的性能优化

探索者是那些寻找不平常的且有用的商业运作模型的用户群。探索者的运作方式是反复无常的、不可预测的及随机的。大部分时间探索者努力搜索,但一无所获。偶尔探索者也会发现意外的、无价的信息"金块"。

探索者查看详细资料和历史记录。在多数情况下,探索者考虑数据的不同类型和数据具体值之间的关系。探索者要做的工作有概括分析、抽取、建模和分类。

(1) 概括分析

概括分析是探索者分析过程的第一步。探索者开始以分析数据仓库中数据的外部特征,即分析数据的完整性和准确性(数据质量)。在概括分析活动中,要询问的典型问题包括:

- 家庭收入如何分配?
- 有多少账户每月消费超过 200 元?有多少账户每月消费小于或等于 200 元?
- 有多少客户有两个以上的小孩并居住在市区?

(2) 抽取

通过概括分析,所选数据的轮廓已经基本显示出来。数据抽取的任务就是从数据仓库中抽取指定的数据,并组织起来,送入支持探索者分析的探索仓库中。这样,不会影响数据仓库的正常工作。

(3) 建模

探索者通过概括分析来理解数据，通过抽取来准备数据，通过建模来分析数据。

建模是开发一种用来描述实体(如客户、商品、渠道等)的关系模型的过程。探索者使用的模型有：

- 客户分段；
- 后续产品；
- 欺诈检测；
- 渠道响应(例如，电话销售和直接邮寄)；
- 信用风险；
- 客户生存期价值；
- 推销响应。

例如，利用建模来确认有可能拖延支付电话账单的客户：首先，建立一个模型(利用统计学和行为科学)来确认经常拖延支付电话账单的客户特征。然后，根据客户与模型的密切程度，对所有的客户分类。这样，可以提供谁将不支付电话账单的某种可能性预测。最后，把那些与此模型有紧密关系的客户作为目标。

数据仓库管理员为保证探索者的有效工作，创建"探索仓库"很有必要。探索仓库是企业数据仓库的"转出"，用来支持某些特定的分析，也不损害企业数据仓库中其他常规用户的正常使用。

建立探索仓库所依赖的技术基础是基于"标识"的技术(参见 4.1.5 小节中 2. 标识技术)，利用基于标识技术可使探索仓库非常经济。基于标识的技术允许把数据压缩到能将数据放置在内存中(全部或者大部分)的程度。一旦使用内存存储，分析和检索的速度将大大快于使用标准企业数据仓库时的速度。

探索仓库是临时性的、短期性的。探索仓库的特征是固定不变的构造和重建。一旦构造好某个探索仓库，则再也不需要构造具有同样形式或内容的探索仓库。探索仓库能够满足数据仓库环境中非结构化处理的需要。探索仓库适合于挖掘数据的探索者。

探索仓库一般使用规范化的数据结构，因为探索仓库适用于不知道自己需求的使用者。而星型模型的数据结构不适合探索仓库，因为星型模型需要知道商业维度(如产品、客户、时间)和指标(如收入或者是成本)等情况下使用数据。

元数据在探索仓库环境中也非常重要。因为探索者用多种方式查看探索仓库，并且有多数方式以前从没有被使用过，所以元数据起到特别重要的作用。在探索仓库中，必须建立有效的元数据层。这个元数据层能够在每次重新构造探索仓库时被传输到探索仓库。

5.1.2 数据管理

数据仓库中存放大量的数据，随时间的延伸又将涌进大量的数据。在数据管理中要处理两大类数据：休眠数据和脏数据。发现这两类数据需要用监视器。

在数据仓库中，不但要对大量存储数据的有效管理，还需要对元数据进行数据管理。

5.1.2.1 休眠数据

1. 休眠数据概念

休眠数据是那些存在于数据仓库中当前不使用、将来也很少使用或不使用的数据。

数据仓库中的数据随着时间的延续，数据被使用的情况会减少，休眠数据随之逐年增加。国外的统计表明：第 1 年内，数据仓库近期数据和综合数据几乎被全部使用。第 2 年内，休眠数据开始出现，数据仓库中的数据有不少未被使用。第 3 年内，休眠数据在增长。第 4 年内，休眠数据迅速增长。

设数据仓库中的数据量为 D，一年之中支持决策的可能的数据处理次数为 n，平均每次处理数据的字节数为 d，则一年中为支持决策的数据处理的总数据量为 nd。

在各次数据处理过程中，可能会出现数据的重复使用，用系数 a 表示为

$$a=\begin{cases}1.0 & \text{如果每次数据处理均没有重复数据}\\ 0.5 & \text{如果平均两次数据处理会遇到同一数据}\\ 0.3 & \text{如果平均三次数据处理会遇到同一数据}\end{cases}$$

则休眠数据量 D_l 表示为

$$D_l = D - and$$

休眠数据占数据仓库中数据的比例称为休眠数据率 R，用公式表示为

$$R = D_l / D$$

休眠数据量 D_l 和休眠数据率 R 只是一个估计数据，实际上休眠数据比估计值可能更高些。对于数据仓库中的休眠数据需要引起数据仓库管理员的重视。

2. 休眠数据的产生与查找

(1) 休眠数据的产生

产生休眠数据的途径有：

① 在数据仓库中输入了过多的近期基本数据。这些过多的数据包含在“事实表”中的列数据和“维表”中的列数据。这些过多的数据在实际使用时并未使用到。

② 过多地增加了不必要的综合数据。数据仓库中为支持用户决策，需要提供多种综合数据，以便使用户能迅速查找到这些综合数据。过多的综合数据会造成浪费，产生休眠数据。

③ 历史数据用于预测，由于过高地估计所需要的历史数据的时间的长度超过预测需求的历史数据，例如，输入了 24 个月的数据，后来发现真正需要的只是 3 或者 4 个月的数据，其他月份数据均是休眠数据，不及时删除，休眠数据会随时间迅速地增长。

(2) 查找休眠数据

查找休眠数据的最好方法是监视用户查询数据仓库的活动。

监视工作包括：

① 监视用户查询的 SQL 语句。

② 监视返回给用户的查询结果数据集。通过监视用户的查询和返回的查询结果，能

够确定查询处理中实际上使用了哪些数据，数据仓库管理员能知道哪些数据没有被使用，它们很可能就是休眠数据。

3. 删除休眠数据

删除休眠数据的方法有以下 3 种。

(1) 直接删除休眠数据

直接删除休眠数据有两种方式：

① 删除用户不访问的数据。这种删除可能会出现的问题是，这些数据将来用户又可能会使用它。同样，最近使用的数据，在将来又可能不使用它。

② 通过数据访问模型来删除休眠数据。数据访问模型是需要和用户一起来确定将来事务活动中需要访问的数据，根据该模型来删除将来不被访问的休眠数据。这种方式更合理。

(2) 对休眠数据归档存储

将已确定的休眠数据归档存入一个大容量的存储媒介中，例如磁带。

(3) 邻线(near line)存储

数据仓库的数据是在线(on line)存储，邻线存储是一种二级数据存储方式。“邻线”介于“在线”和“离线(off line)”之间，将休眠数据从数据仓库的在线存储中转移到邻线存储中，平时不参与数据仓库的运行，但在必要时，可以被在线存储合理利用。邻线存储的花费比在线存储少，但比归档存储多。这是一种有效的删除休眠数据的方式。

5.1.2.2 脏数据的产生和清理

脏数据是指在数据源中抽取、转换和装载到数据仓库的过程中出现的多余数据和无用数据。

1. 产生脏数据的途径

(1) 开始时定义了一些多余的数据或由于一些不合适的转换规则在转换过程中产生的无用数据。

(2) 来自不同数据源的数据在数据结构、数据编码、数据定义等方面是不兼容的，在集成这些数据时未对所有不同情况的数据都转换成统一形式，产生遗漏或用了不匹配的转换方法而产生脏数据。

(3) 输入的数据已经过期。由于工作业务的改变，某些前期业务的数据已经过期，仍遗留在数据仓库中而造成的过期无用数据。

(4) 用户需求的改变或数据质量有了新的要求时，那些没有适应改变要求的数据成了无用的脏数据。如刷新数据的周期缩短后，未适应刷新要求的旧数据已成为无用数据。

2. 清理脏数据

清理以上脏数据的方法有：

（1）检查抽取数据的定义和数据转换规则的正确性，对那些不合适的定义与规则所造成的脏数据进行清理。

（2）在对多个数据源进行集成时，必须对所有的不同结构、不同编码、不同定义的数据，严格按统一形式转换后再集成，清除那些遗漏或不匹配方法而产生的脏数据。

（3）对过期数据，在形成历史数据后，根据这种数据量的大小来决定是否需要进行重新整理：对数据量较少时进行重新整理；对数据量较大时，增加一些时间限制的规则来帮助对数据的使用。

5.1.2.3 监视数据

管理大量数据的最好方法是删除休眠数据和脏数据。为了删除休眠数据和脏数据，必须查找哪些数据是真正休眠数据和脏数据，识别它们的最好方法是利用活动监视器（即数据使用跟踪器）。活动监视器位于最终用户和数据仓库之间并且查看每个经过系统的活动，输入数据仓库中的 SQL 语句和查询活动产生的结果集可以通过监视器来检查。监视活动分为 3 个级别：表格级、表格/列级和表格/列/值级。监视活动的开销一般是较大的。

1. 监视休眠数据

监视休眠数据分为三级：表格级休眠、列级休眠和值级休眠。

（1）表格级休眠

它发生在没有使用某个实体中的一个表格时。表格休眠通常出现于小表格和包括概括数据的表格中，这些类型的表格通常是在数据临时被使用时创建的。在分析结束时，忘记了清除系统中这些普通创建的作业表格。另一种可能性是系统在需要的时候创建概括表格，但后来再也不需要这些表格。当发现这些表格后，从数据仓库中删除实体中的这个表格即可。

（2）列级休眠

列级休眠出现在当某一整列或多列不被访问时。造成的原因是最终用户没有真正认识到某列在将来会有什么用途，但在设计数据仓库时却指定了需要此列。由于数据仓库要承担未知的探索，那么出现这样的一个需求并没有什么不合理。出现列休眠的另一个原因是最终用户不知道这列存在。或许最终用户发现了一个更好的数据源可用来代替此列。删除休眠列不是一件容易的事情，对这类数据的重组需要大量资源。

（3）值级休眠

数据休眠出现在某个表格中的数值不会被访问时。这是一种非常普遍的数据休眠类型。产生的原因是最终用户指定了过多的大量历史数据。例如用户表示需要两年的数据，一旦建立数据仓库并加载了两年的历史数据，结果用户发现实际上只需要 3 个月的数据，造成大量历史数据呆在数据仓库中而没有什么用途。造成数据值休眠的另一种原因是这类数据毫无商业利益。

一旦发现数据值休眠就删除它是非常容易的事。识别出这种值就从数据仓库中读取它，然后删除或归档。最后由数据仓库管理员（DBA）回收它们占据的空间。

2. 监视脏数据

活动监视器监视数据仓库中的数据内容，并且数据业务规划识别出不符合规划的所有行或数据记录。

当监视器发现脏数据时，是否意味着必须立刻改正数据？答案是，但不是必需的。在很多方面，在数据仓库中监视数据所带来的问题比它能解决的问题更多。

(1) 如果只有少量不正确的记录，有可能并不值得做这种改正工作。

(2) 如果有大量的数据是不正确的，则补救的方法是改正数据之前，修改引起这种问题的程序。

(3) 某些时候，虽然知道数据不正确，但是没有办法知道合适的数值应该是多少。

在多数时候，只是需要获得脏数据量，而不需要实际进入并改正数据。

5.1.2.4 元数据管理

1. 评估元数据的价值

元数据在多个级别上为数据仓库创造价值。

(1) 描述应用程序操作数据的机制和控制运行机制的元数据，使系统开发人员就能够理解应用程序内部结构和数据之间的相互关系。

(2) 在数据仓库环境中元数据通过 3 种方式发挥作用：

- 描述源和目标的数据模型；
- 在装载数据时描述转换集成的数据流；
- 为用户导航，找到所需要的数据。

(3) 获取数据和使用数据的元数据是元数据价值最大的用途。一个好的数据仓库具备从多个系统中合并数据和使用数据的能力，包括数据仓库设计和建模工具，数据抽取工具、转换、合并、清洗工具，查询分析和执行管理工具，终端用户查询和分析工具等。为了使这些工具有效地协同工作，这些工具必须可以共享在这个环境中共同感兴趣的元数据。

随着技术的发展，元数据访问和协同工作越来越重要。下一代数据仓库已经从大的数据仓库体系结构开始过渡到小的、分布式的、面向特定应用的数据集市。

在分布式环境中，多个数据集市致力于特定的面向商业功能单元的决策支持需求，如销售、金融、产品管理、售后服务等。这些分布式的数据集市既可以是关系数据库(一般采用星型结构)，也可以是多维立方体，能够分散生成各自的子决策支持系统。

2. 管理元数据

随着元数据越来越成为公司重要的资源，越来越需要健壮的元数据管理功能，包括：

(1) 支持企业范围内的体系结构

企业在开发应用程序使用数据仓库时，企业关心软件设计与开发、用户接口、操作管理、应用程序内部的消息传递、数据的协同工作能力。所有这些都驱使开发人员去理解各种元数据内容，以及元数据在企业范围内的作用。

(2) 基于知识库的方法

元数据一般存储在其特定的元数据知识库中。因此,企业可以要求提供一种机制,可以将不同工具支持的元数据无缝地转移到一个共享的、公共的元数据知识库中。

(3) 配置管理

元数据知识库必须提供标准的配置管理能力,如注册、退出、版本控制等。还需要提供抽取、修改元数据的定义以及将其定义存到知识库中,还必须具有在必要的时候将元数据恢复到某一个前版本的功能。

(4) 支持开放的元数据交换标准

企业内部和外部对元数据的访问导致了对开放的元数据交换标准支持的需求。至少,企业元数据应该支持 MDIS(元数据交换标准)。

(5) 动态交换和同步

企业应该采用 MDIS 标准,实现动态交换或同步,否则需要一个开放的元数据交换工具。

5.2 数据仓库的决策支持与决策支持系统

数据仓库是一种能够提供重要战略信息,并获得竞争优势的新技术,从而得到迅速的发展。

经理们和管理者需要哪些战略信息来支持决策呢?例如,对自己公司的运营有全面深入的了解,了解关键因素和它们之间是如何相互作用的;监视这些因素是如何随时间变化的;将公司的运营状况和市场竞争及行业标准联系起来比较多。经理们和管理者需要将注意力集中在客户的需求和喜好上,集中在新兴技术、销售、市场结果、产品和服务质量水平等事务上。制定和执行商业战略及目标时需要的信息类型应包含整个企业组织。

战略信息并不为企业日常运作所用,不是关于订货、发货、处理投诉或者从银行账户提款的信息。战略信息比这些信息重要得多,对于企业的生存和持续健康发展有非常重要的意义。企业决定性的商业决策有赖于正确的战略信息。

具体的战略信息有:

(1) 给出销售量最好的产品名单;

(2) 找出出现问题的地区;

(3) 追踪查找出现问题的原因(向下钻取);

(4) 对比其他的数据(横向钻取);

(5) 显示最大的利润;

(6) 当一个地区的销售低于目标值时,提出警告信息。

建立数据仓库的目的不只是为了存储更多的数据,而是要对这些数据进行处理并转换成商业信息和知识,利用这些信息和知识来支持企业进行正确的商业行动,并最终获得效益。

数据仓库的功能是在恰当的时间,把准确的信息传递给决策者,使决策者能作出正确的商业决策。

数据仓库的主要作用是帮助企业摆脱盲目性，提高决策的准确性和决策速度，也就是说，数据仓库的作用正是帮助企业把信息与知识转变为力量（实施正确的行动并获得效益）。

数据仓库的决策支持一般包括查询与报表、多维分析与原因分析、预测未来。NCR 数据仓库公司提出了动态数据库及相应的决策支持：实时决策和自动决策。

针对实际问题，利用决策支持能力，通过人机交互，达到辅助决策的系统称为决策支持系统。

5.2.1 查询与报表

查询与报表是数据仓库的最基本、使用最多的决策支持方式。通过查询与报表使决策者了解目前发生了什么。

1. 查询

数据仓库提供的查询环境的特点是：

（1）能向用户提供查询的初始化、公式表示和结果显示等功能。

（2）由元数据来引导查询过程。

（3）用户能够轻松地浏览数据结构。

（4）信息是用户自己主动索取的，而不是数据仓库强加给他们的。

（5）查询环境必须要灵活地适应不同类型的用户。

查询服务具体体现为：

（1）查询定义。确保数据仓库用户能够容易地将商业需求转换成适当的查询语句。

（2）查询简化。让数据和查询公式的复杂性对用户透明。让用户能够简单地查看数据的结构和属性。使组合表格和结构简单易用。

（3）查询重建。有些简单的查询也能导致高强度的数据检索和操作，因此要使用户输入的查询进行分解并重新塑造，使其能更高效地工作。

（4）导航的简单性。用户能够使用元数据在数据仓库中浏览数据，并能容易地用商业术语而不是技术术语来导航。

（5）查询执行。使用户能够在没有任何 IT 人员的帮助下提高并执行查询。

（6）结果显示。能够以各种方法显示查询结果。

（7）对聚合的了解。查询过程机制必须知道聚合的事实表，并且在必要的时候能够将查询重新定义到聚合表格上，以加快检索速度。

2. 报表

大部分查询均要以报表形式输出。数据仓库构建的报表环境有：

（1）预格式化报表。提供这些报表清晰的描述说明。使用户能够容易地浏览格式化报表库中的报表并选择需要的报表。

（2）参数驱动的预定义报表。与预格式化报表相比，参数驱动的预定义报表给了用户更多的灵活性。用户必须有能力来设置自己的参数，用预定义格式创建报表。

（3）简单的报表开发。当用户除了与格式化报表或预定义报表外还需要新的报表时，

必须能够轻松地利用报表语言撰写工具来开发自己的报表。

（4）公布和订阅。数据仓库设置选项让用户公布自己创建的报表，并允许其他用户订阅或者接收这些报表的复制。

（5）传递选项。提供各种选项，诸如群发、电子邮件、网页和自动传真等，让用户传递报表，允许用户选择自己的方法来接收报表。

（6）多数据操作选项。用户可以请求获得计算出来的指标，通过交换行和列变量来实现结果的旋转，在结果中增加小计和最后的总计，以及改变结果的排列顺序等操作。

（7）多种展现方式选项。提供多种类型的选项，包括图表、表格、柱形格式、字体、风格、大小和地图等。

5.2.2　多维分析与原因分析

多维分析与原因分析能让决策者了解"为什么会发生"。

1. 多维分析

多维分析是数据仓库的重要的决策支持手段。数据仓库中心数据是以多维数据存储的。通过多维分析将获得在各种不同维度下的实际商业活动值（如销售量等），特别是它们的变化值和差值，达到辅助决策效果。例如通过多维分析得到如下信息：

- 今年以来，公司的哪些产品量是最有利润的？
- 最有利润的产品是不是和去年一样？
- 公司今年这个季度的运营和去年相比情况如何？
- 哪些类别的客户是最忠诚的？

这些问题的答案是典型的基于分析的面向决策的信息。决策分析往往是事先不可知的。例如，一个经理可能会以查询品牌利润，按地区的分布情况来开始他的分析活动。每一个利润的数值指的是，在指定的时间内，某个品牌所有产品在该地区的所有地方销售利润的平均值。每一个利润数值都可能是由成千上万的原始数据汇聚而成的。

这些分析都是建立在多维数据分析之上进行的。

2. 原因分析

查找问题出现的原因是一项很重要的决策支持任务，一般通过多维数据分析的钻取操作来完成。

某公司从分析报表中得知最近几个月来整个企业的利润在急速下滑，为此系统分析员利用数据仓库的原因分析的决策支持手段，通过人机交互找出该企业利润下滑的原因。具体步骤如下：

（1）查询整个公司最近3个月来各个月份的销售额和利润，通过检索数据仓库中的数据显示销售额正常，但利润下降。

（2）查询全世界各个区域每个月的销售额和利润，通过检索多维数据和切块，显示欧洲地区销售额下降，利润急剧下降，其他地区正常。

（3）查询欧洲各国销售额和利润。通过对多维数据的钻取，显示一些国家利润率上升，

一些国家持平，欧盟国家利润率急剧下降。

(4) 查询欧盟国家中的直接和间接成本。通过对多维数据的钻取，得到欧盟国家的直接成本没有问题，但间接成本提高了。

(5) 查询间接成本的详细情况。通过钻取查看详细数据，得出企业征收了额外附加税，使利润下降。

通过原因分析，得到企业利润下滑的真正原因是欧盟国家征收了额外附加税而造成的。

在数据仓库中，对宏观数据中发现的问题，通过向下钻取操作，查看下层大量详细的多维数据，才能发现问题出现的原因。针对具体问题，通过数据仓库的原因分析，找出问题发生的原因过程，这是一个典型的数据仓库决策支持系统简例。

5.2.3 预测未来

预测未来使决策者了解“将要发生什么”。

数据仓库中存放了大量的历史数据，从历史数据中找出变化规律，将可以用来预测未来。在进行预测的时候需要用到一些预测模型。最常用的预测方法是采用回归模型，包括线性回归或非线性回归。利用历史数据建立回归方程，该方程代表了沿时间变化的发展规律。预测时，代入预测的时间到回归方程中去就能得到预测值。一般的预测模型有多元回归模型、三次平滑预测模型、生长曲线预测模型等。

除用预测模型外，采用聚类模型或分类模型也能达到一定的预测效果。

聚类模型是对没有类的大量实例，利用距离的远近（如欧式距离和海明距离等），把大量的实例聚成不同的类，如 K-means 聚类算法和神经网络的 Kohonen 算法等。把实例聚完类后，对新的例子，仍用距离大小来判别它属于哪个类。

对于分类模型，它是对已经有了类别后，分别对各个不同类进行类特征的描述，如决策树方法、神经网络的 BP 模型等。分类模型是通过对各类实例的学习后，得到各类的判别知识（即决策树、神经网络的网络权数值等），利用这些知识可以对新例判别它属于哪个类别。

5.2.4 实时决策

数据仓库的第 4 种决策支持是企业需要准确了解“正在发生什么”，从而需要建立动态数据仓库（实时数据库），用于支持战术型决策，即实时决策，有效地解决当前的实际问题。第 1 到第 3 种决策支持的数据仓库都以支持企业内部战略性决策为重点，帮助企业制定发展战略。数据仓库对战略性的决策支持是为企业长期决策提供必需的信息，包括市场划分、产品（类别）管理战略、获利性分析、预测和其他信息。战术性决策支持的重点则在企业外部，支持的是执行公司战略的员工。第 4 种侧重在战术性决策支持。

数据仓库的“实时决策”是指为现场提供信息实时支持决策，如能及时补给的库存管理和包裹发运的日程安排及路径选择等。许多零售商都倾向于由卖主管理库存，自己则拥有一条零售链和众多作为伙伴的供货厂商，其目的是通过更有效的供货链管理来降低库存成本。为了使这种合作获得成功，就必须向供货商详细地提供有关销售、促销推广、库内存货等信息的知情权。之后便可以根据每个商店和每个商品对库存的要求，建立并实施有效的生产和交货计划。为了保证信息确实有价值，必须随时刷新信息，还要非常快地对查询作出

响应。

动态数据仓库能够逐项产品、逐个店铺、逐秒地作出最佳决策支持。

以货运为例。统筹安排货运车辆和运输路线,需要进行非常复杂的决策。卡车上的货物常常需要打开,把某些货物从一辆车转移到另一辆车上,以便最终送抵各自的目的地。这有些像旅客在枢纽机场转机。当某些卡车晚点时,就要作出艰难的决定:是让后继的运输车等待迟到的货物,还是让其按时出发。如果后继车辆按时出发而未等待迟到的包裹,那么迟到包裹的服务等级就会大打折扣。反过来说,等待迟到的包裹则将损害在后继的运输车上的其他待运包裹的服务等级。

运输车究竟等待多长时间,取决于需卸装到该车辆的所有延迟货物的服务等级和已经装载到该车辆的货物的服务等级。很显然,第二天就应该抵达目的地的货物和数天后才需抵达目的地的货物,二者的服务等级及其实现难度是大不相同的。此外,发货方和收货方也是决策的重要考虑因素。对企业赢利十分重要的客户,其货物的服务等级应该相应提高,以免因货物迟到破坏双方关系。延误货物的运输路线、天气条件和许多其他的因素也应予以考虑。能够在这种情况下作出明智的决策,相当于解决了一个非常复杂的优化问题。

显然易见,零担散货部经理应在先进决策支持功能的帮助下,极大地提高其计划和路径选择的决策质量。更重要的是,若要实现数据仓库的决策支持能力,作为决策基础的信息就必须保持随时更新。这就是说,为了使数据仓库的决策功能真正服务日常业务,就必须连续不断地获取数据并将其填充到数据仓库中。战略决策可使用按月或周更新的数据,而以这种频率更新的数据是无法支持战术决策的。此外,查询响应时间必须以秒为单位来衡量,才能满足作业现场的决策需要。

与传统的数据仓库一样,最佳的动态数据仓库是跨越企业职能和部门界限的。它既可为战术决策也可为战略决策提供资源支持。动态数据仓库是为支持企业级业务目标而设计的。与传统的数据仓库相比,更加深入到企业内部,能将企业的多种渠道,包括网络、呼叫中心和其他客户联络点联为一体,还意味着通过网络,在企业各个角落配置决策人员。

动态数据仓库的主要功能是缩短重要业务决策及其实施之间的时间。重要的是将动态数据仓库所做的数据分析转换成可操作的决策,这样才能将数据仓库的价值最大化。动态数据仓库的主导思想是提高业务决策的速度和准确性,其目标是达到近乎实时决策,生成最大价值。

5.2.5 自动决策

数据仓库的第 5 种决策支持是由事件触发,利用动态数据库自动决策,达到“希望发生什么”。

动态数据仓库在决策支持领域中的角色越重要,企业实现决策自动化的积极性就越高。在人工操作效果不明显时,为了寻求决策的有效性和连续性,企业就会趋向于采取自动决策。在电子商务模式中,面对客户与网站的互动,企业只能选择自动决策。网站中或 ATM 系统所采用的交互式客户关系管理(CRM)是一个个性化产品供应、定价和内容发送的优化客户关系的决策过程。这一复杂的过程在无人介入的情况下自动发生,响应时间以秒或毫秒计算。

随着技术的进步，越来越多的决策由事件触发，自动发生。例如，零售业正面临电子货架标签的技术突破。该技术的出现废除了原先沿用已久的手工更换的老式聚酯薄膜标签。电子标签可以通过计算机远程控制，改变标价，无需任何手工操作。电子货架标签技术结合动态数据仓库，可以帮助企业按照自己的意愿，实现复杂的价格管理自动化；对于库存过大的季节性货物，这两项技术会自动实施复杂的降价策略，以便以最低的损耗售出最多的存货。降价决策在手工定价时代是一种非常复杂的操作，往往代价高昂，超过了企业的承受能力。带有促销信息和动态定价功能的电子货架标签，为价格管理带来了一个全新的世界。而且，动态数据仓库还允许用户采用事件触发和复杂决策支持功能，以最佳方案，逐件货品，逐家店铺，随时作出决策。在 CRM 环境中，利用动态数据仓库，根据每一位客户的情况作出决策都是可能的。

激烈的竞争形势和日新月异的技术革新推动了决策技术的进步。动态数据仓库可以为整个企业提供信息和决策支持，而不只限于战略决策过程。然而，战术决策支持并不代替战略决策支持。确切地说，动态数据仓库同时支持这两种方式。动态数据仓库的主要工作量仍然是战略性的。

5.2.6 决策支持系统

数据仓库整合了企业的各种信息来源，能确保一致与正确、详细的数据。它是一个庞大的数据资源。要将数据转换成商业智能，就需要利用数据仓库来建立决策支持系统。

基于数据仓库的决策支持系统是针对实际问题，利用分析工具或者编制程序，采用一种或多种组合的决策支持能力，例如随机查询、灵活的报表、预测模型等，对数据仓库中的数据进行多维分析，从而掌握企业的经营现状，找出现状的原因，并预测未来的发展趋势，弥补经验和直觉的不足，协助企业制定决策，增强竞争优势。

根据 NCR 公司在企业政策制定调查中，发现企业的决策危机日益严重。虽然有更多的数据，但是也有更多的决策，同时决策也更加复杂化。

调查中有 98%的管理者说数据一直在增加中，随着数据每年 2 倍或 3 倍的增长，他们会被数据“淹没”。有 75%的管理者表示他们每天所做的决策比以往多。有 52%的决策更为复杂，这其中有 83%的人说他们必须针对每一决策去咨询 3 个或更多的信息来源。

只有建立基于数据仓库的决策支持系统，才能适应这种发展趋势，才能在适当的时间获得正确的信息，快速地将这些信息转换成正确的决策。

NCR 公司总裁 M. Hard 列举了 3 个不同性质的公司失败的案例，是不明智决策的结果。

(1) 霸菱银行，英国最老的银行之一(成立于 1762 年)，在 1995 年因为在新加坡分公司一位员工有 29000 美元的错误，在伦敦的管理层，并不清楚在新加坡所发生的状况，由于在决策上历经一连串错误的决策，不出 3 年，银行垮了。分析原因：霸菱银行缺乏企业单一整合的观点，缺乏可用详细的数据，显然在每日、每周甚至于每年的基准上，缺乏适当的检查点或事业监督。

(2) F. W. Woolworth 于 1879 年在美洲开了第一家店，118 年来它提供了优惠价格的产品，培养了广大的客户忠诚度。它一直是人们的采购商品的地方，可买到任何东西。但

是，他忽略了人口统计的改变与人们搬住郊区的趋势，未实时随市场的改变而调整，最终被崭新的零售业，如 Wal-Mart 与 Target 等公司击败。

(3) 美国环球航空 TWA，1920 年开始航空邮递时代，在 1930 年，它在现代技术进展上领先，曾横贯大陆与横贯大西洋的飞行。但是，后来它缺乏信息科技的基础建设来应付新的竞争环境，在多处还停留在 30 年前技术的基础建设上，在倒闭前一年，终于了解必须结合来自多个系统的财务、市场与销售数据，以便因市场改变快速而作出精确的反应，但一切都为时过晚。

对以上的 3 个公司的分析得出，建立基于数据仓库的决策支持系统的公司也许可以避免失败的命运。

5.3 数据仓库应用实例

5.3.1 航空公司数据仓库决策支持系统简例

1. 航空公司数据仓库系统的功能

航空公司数据仓库功能模块有如下几个。

- 市场分析：分析国内、国际、地区航线上的各项生产指标。
- 航班分析：分析某个特定市场上所有航班的生产情况。
- 班期分析：分析某个特定市场上各班期的旅客、货运分布情况。
- 时段分析：分析一段时间范围内每天不同时段的流量分布。
- 效益分析：分析航线、航班的效益。
- 机型分析：分析不同种机型对客座率等关键指标的影响。
- 因素分析：分析某个关键指标发生变化后对其他指标的影响程度。

2. 数据仓库系统的决策支持

利用数据仓库系统提供的决策支持有：

- 一段时间内某特定市场占有率、同期比较、增长趋势；
- 各条航线的收益分析；
- 计划完成情况；
- 流量、流向分析；
- 航线上各项生产指标变化趋势的分析；
- 航线上按班期分析、汇总各项趋势；
- 航线上按航班时刻分析各项指标；
- 航线上不同航班性质比较；
- 航线上运力投入结构比较；
- 分机型的航线运输统计；
- 飞机利用率统计；

- 城市对流量、流向对比；
- 航向分机型收益比较；
- 航班计划评估；
- 航线上不同机型的舱位利用情况。

3. 决策支持系统简例

通过查询“北京到各地区的航空市场情况”，发现西南地区总周转量出现了最大负增长量。该决策支持系统简例就是完成对此问题进行多维分析和原因分析，找出出现的原因。

具体步骤如下：

(1) 查询：全国各地区的航空总周转量并比较去年同期状况

从数据仓库的综合数据中查出北京到国内各地区航空周转量并与去年同期比较增长量，制成直方图进行显示，如图 5.1 所示。

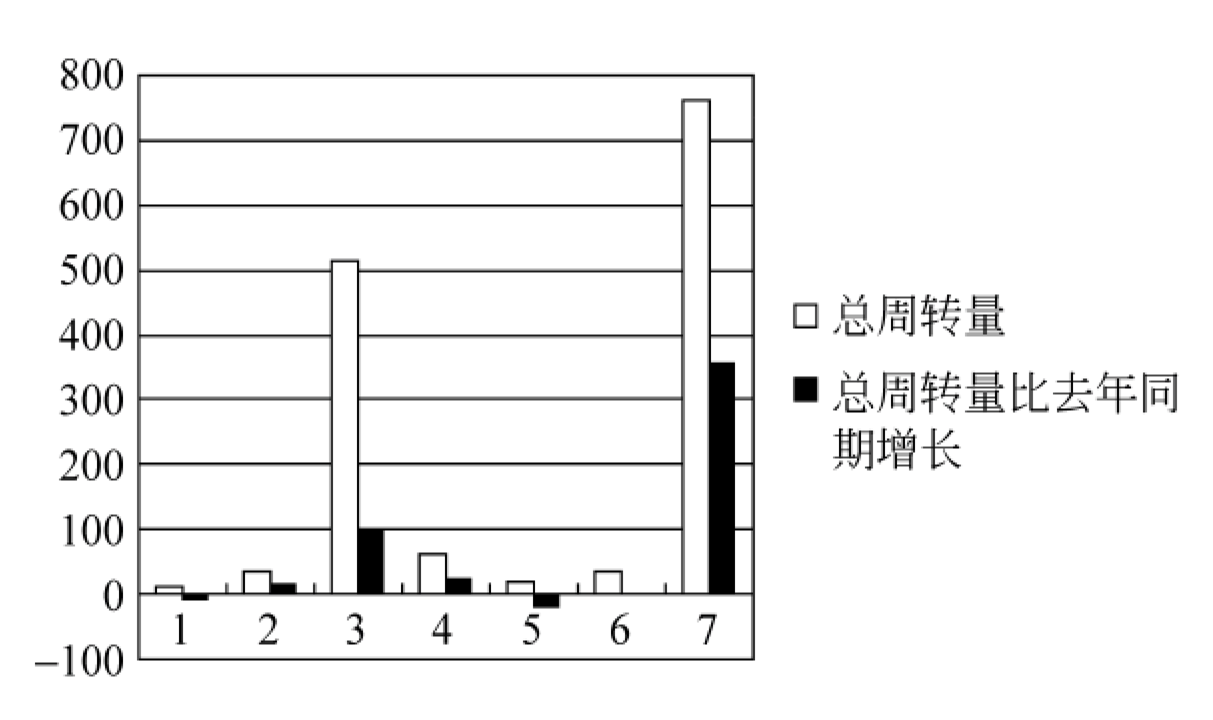


图 5.1　全国各地区航空周转量与去年对比状况

从图 5.1 中看到从北京到国内各地区的总周转量以及与去年同期的比较情况，发现“北京—西南地区”出现的负增长最大。

(2) 查询：全国各地区客运周转量以及与去年同期相比较

从数据仓库的总周转量数据中下钻到客运周转量并与去年同期比较增长量，制成直方图显示，如图 5.2 所示。

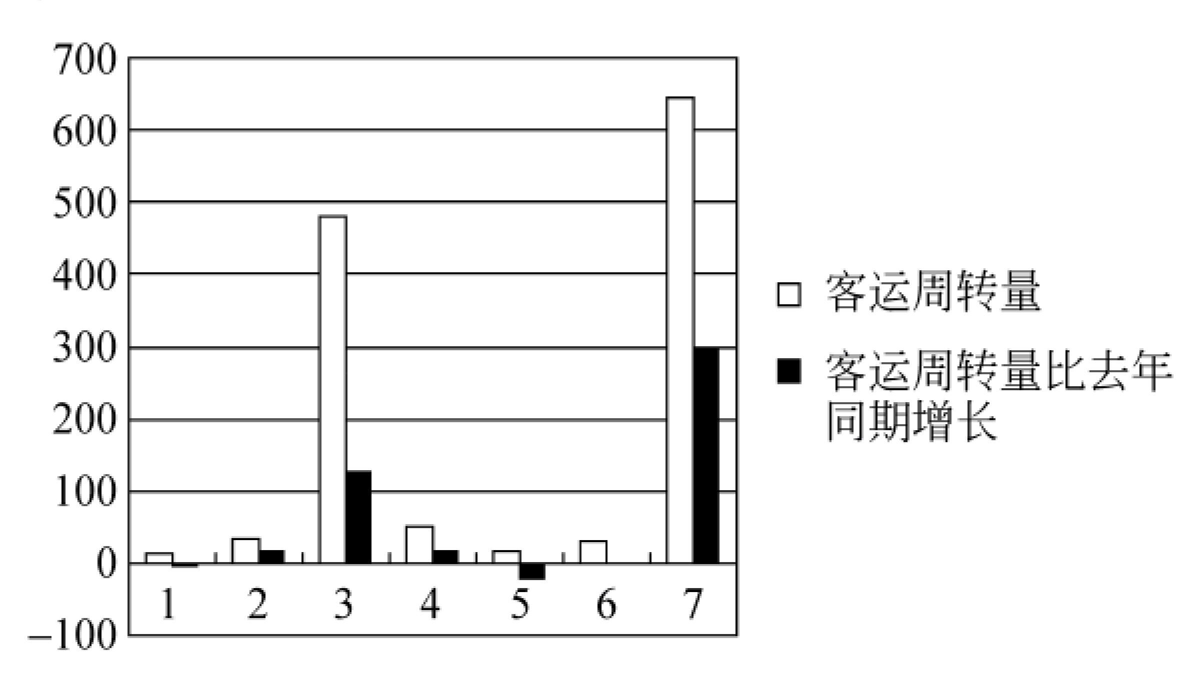


图 5.2　全国各地区航空客运周转量及与去年同期比较

从图 5.2 中看到客运周转量及与去年同期比较，西南地区负增长在全国是最大的，其次是东北地区。

(3) 查询：全国各地区航空货运周转量及与去年同期比较

从数据仓库的总周转量数据中下钻到货运周转量并与去年同期比较增长量，制成直方图显示，如图 5.3 所示。

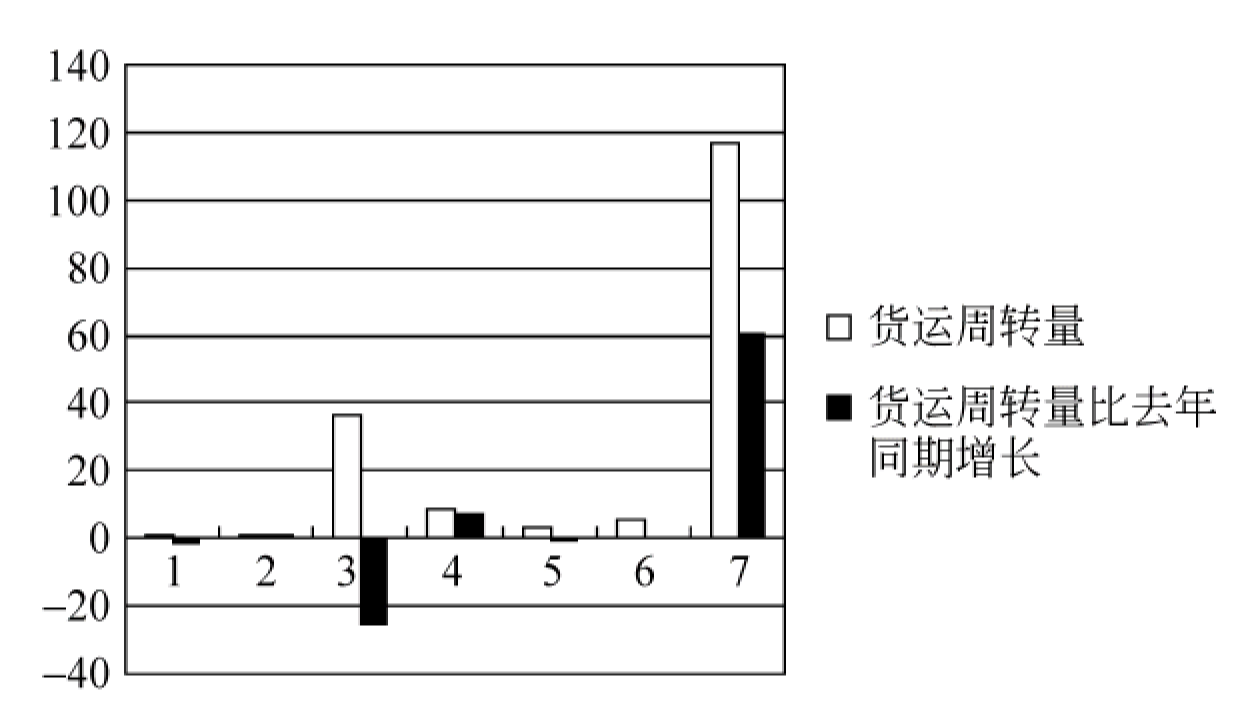


图 5.3　北京到国内各地区货运周转量及与去年同期比较

从图 5.3 中看到货运周转量及与去年同期比较，华东地区负增长在全国是最大的，西南地区也有负增长。

(4) 查询：全国各地区客运、货运、总周转量及与去年同期比较的具体数据

从数据仓库综合数据中直接取数据，制成表格显示，如表 5.1 所示。

表 5.1　客运、货运、总周转量及与去年同期比较

项目	客运周转量	对比去年增长量	货运周转量	对比去年增长量	总周转量	对比去年增长量
东北地区	11.86	−5.1	1.29	−1.5	13.15	−6.6
华北地区	34.88	15.03	1.11	0.75	36	15.78
华东地区	479.30	126.52	36.16	−25.59	515.46	100.93
西北地区	51.60	18.05	9.0	7.2	60.6	25.25
西南地区	15.43	−19.35	3.29	−0.56	18.72	−19.91
新疆地区	29.02	0	5.85	0	34.87	0
中南地区	643.43	295.86	116.85	60.70	760.28	356.56

从表 5.1 中可以看出航空客运、货运、总周转量以及与去年同期比较的具体数据。西南地区总周转量的负增长主要是客运负增长为主体。

(5) 查询：西南地区昆明、重庆两地航空总周转量以及与去年同期比较

从数据仓库总周转量下钻到西南地区昆明、重庆两地的总周转量以及与去年同期的比较，制成直方图显示，如图 5.4 所示。

从图 5.4 中看出，西南地区航空总周转量下降最多的是昆明航线。

(6) 查询：昆明航线按不同机型显示各自的总周转量并比较去年同期情况

从数据仓库中西南地区取出按机型维的各自机型的总周转量以及比较去年同期增长

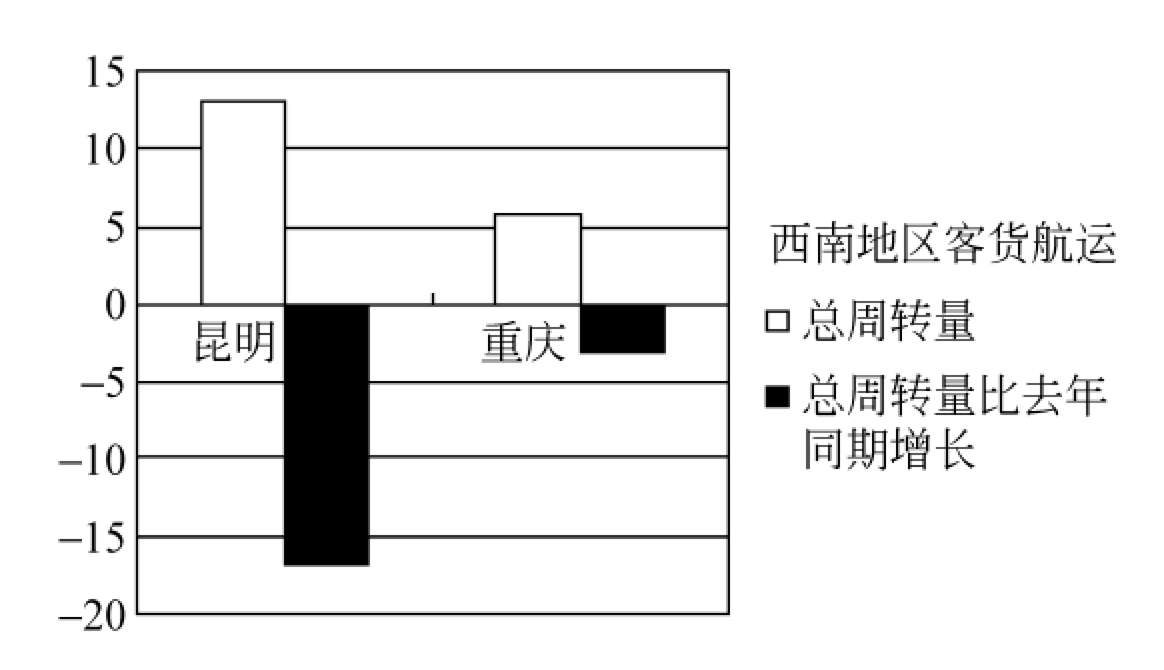


图 5.4　西南地区昆明、重庆两地航空总周转量及与去年同期比较

量,用柱形图显示,如图 5.5 所示。

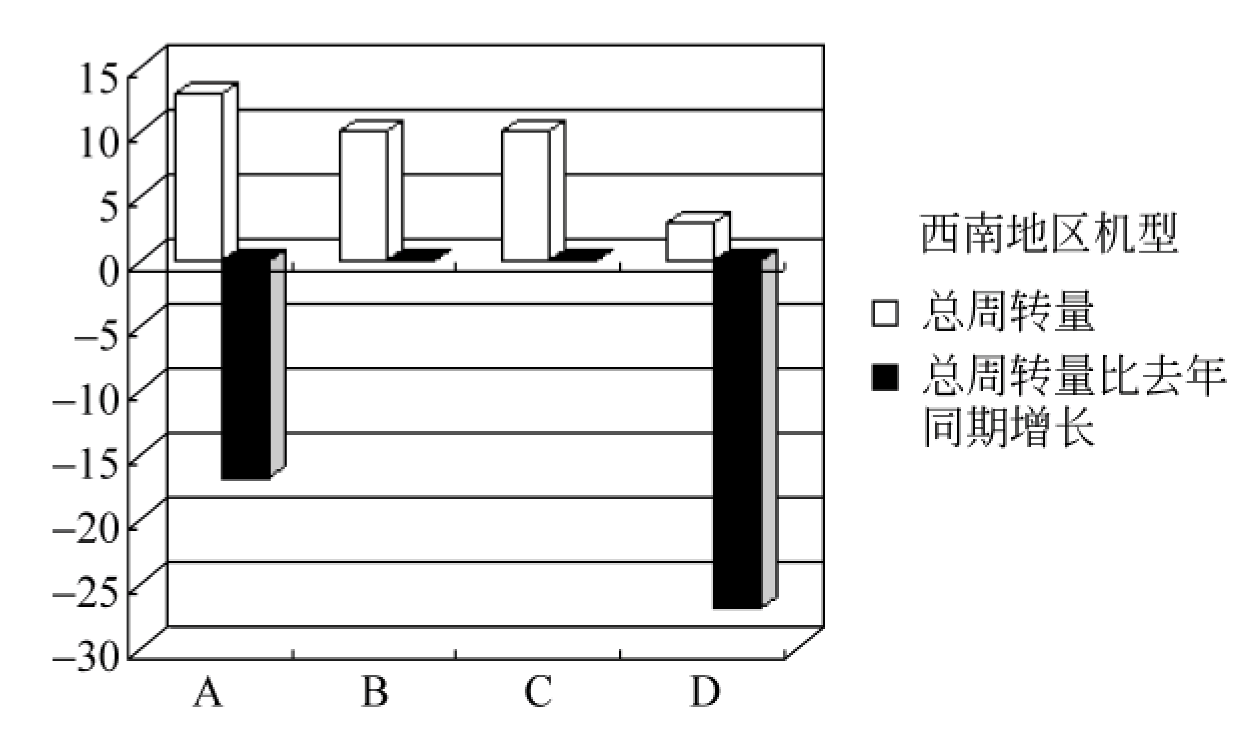


图 5.5　昆明航线各机型总周转量以及与去年同期比较的柱形图

从图 5.5 可以看出昆明航线中 200～300 座级机型负增长最大,其次是 150 座级机型也有较大的负增长,而 200 座级以及 300 座级以上机型保持同去年相同航运水平。

(7) 查询:昆明航线按不同机型的周转量并比较去年同期的具体数据

从数据仓库中直接取数据,制成表格显示,如表 5.2 所示。

表 5.2　昆明航线各机型总周转量以及与去年同期比较的数据

项目	总周转量	对比去年增长量
150 座级	12.99	－16.83
200 座级	10.07	0
300 座级以上	10.07	0
200～300 座级	2.91	－26.9

从表 5.2 中可以看出,不同机型的总周转量以及对比去年同期增长的具体数据。

以上决策支持系统过程完成了对航空公司全国各地区总周转量对比去年同期出现负增长量最大的西南地区,经过多维分析和原因分析,找出其原因发生在昆明航线上,主要是 200～300 座级机型的总周转量负增长以及 150 座级机型负增长量造成的。其中,200～300 座级负增长最严重。这为决策者提供了解决西南地区负增长问题辅助决策的信息。

4. 决策支持系统结构图

将以上决策支持系统过程用决策支持系统结构图画出,如图 5.6 所示。

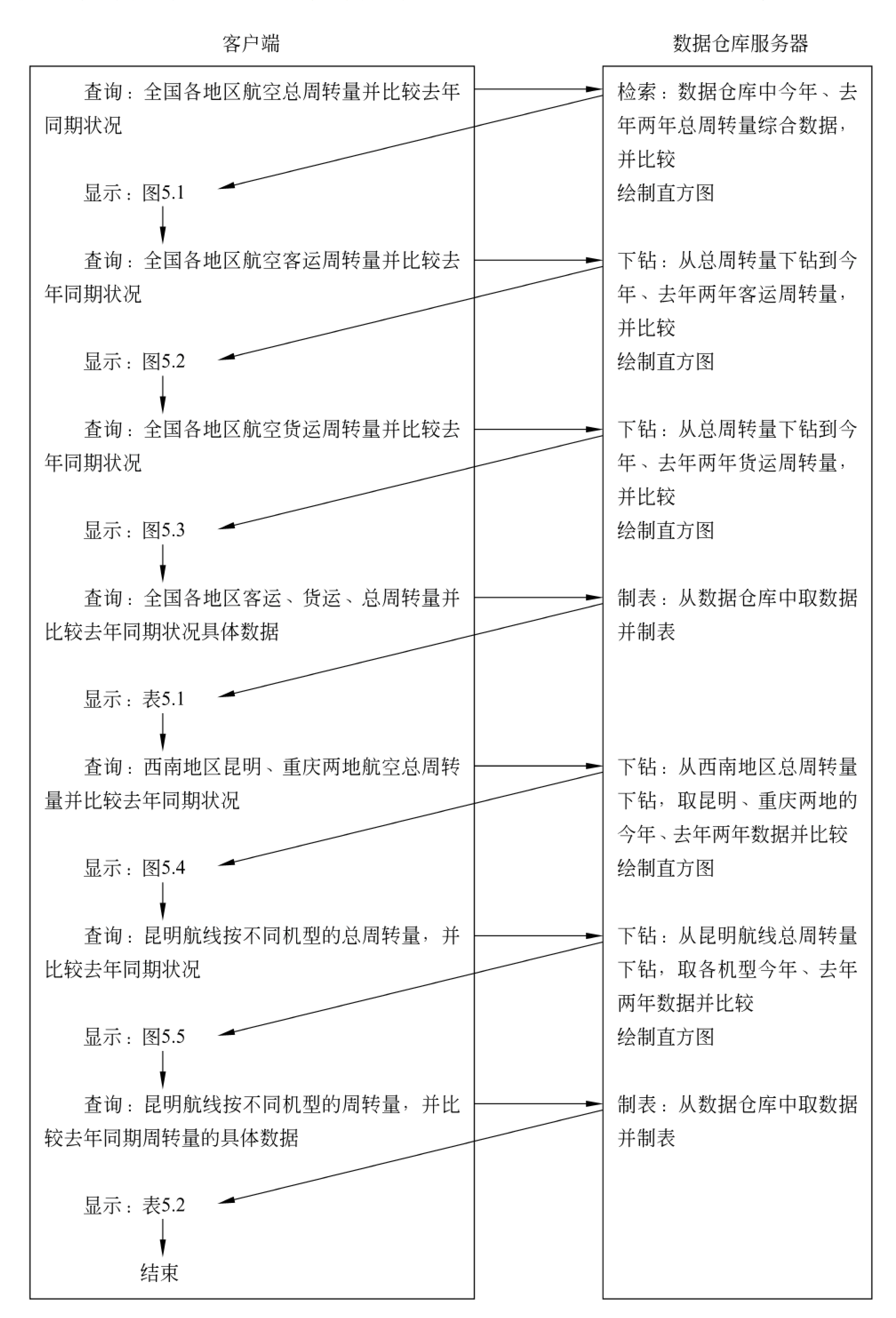


图 5.6　决策支持系统结构图

5. 决策支持系统应用

以上决策支持系统只是找出西南地区航运负增长问题是由于在昆明航线上200～300座级以及150座级机型的负增长所直接造成的原因之一。还可以通过昆明航线上航班时间以及其他方面进行原因分析，找出其他原因，为决策者提供更多的辅助决策信息。

同样，可以从国内各地区航空市场状况中对比去年同期增长显著的中南地区，找出总周转量大幅提高的原因。

从正反两方面进行多维分析和原因分析，将得到更多的辅助决策信息，减少负增长，增大正增长，以提高更大利润。

进行多方面分析的大型决策支持系统将可以发挥更大的辅助决策效果。

5.3.2 统计业数据仓库系统

1. 统计业数据仓库解决方案

统计信息是科学决策和宏观管理的重要基础，是国民经济核算的中心，是了解国情国力、指导国民经济和社会发展的信息主体。统计部门作为国家法定的专职信息职能部门，担负着对国民经济和社会发展情况进行统计调查、统计分析、提供统计资料和统计咨询意见、实行统计监督的神圣职责。

中国的统计事业近年来得到了长足的发展，统计服务水平也有了显著的提高，人们对统计的认识也进一步加强。但同时，中国的统计工作也面临进一步的改革。一方面，社会主义市场经济的迅速发展，经济体制与经济增长方式的逐步转变，对统计工作提出了更新更高的要求；另一方面，计算机和通信技术的迅速发展带来了全球信息化革命，新的信息技术已经并正在引起统计设计、数据处理、信息管理与服务技术的重大变革。

目前，国外统计行业成功的做法之一是采用先进的、成熟的数据仓库技术。数据仓库是信息技术领域的新概念，是近年来迅速发展起来的一种信息存储及管理技术。它存储大量的、决策分析所必需的、历史的、分散的各种数据，经过处理将这些资料和数据转换成集中统一、随时可用的信息。它能方便地提供统计业务人员和各级领导进行随机查询和任意的分析处理；它具有在任何时间、任何业务、回答任何问题的能力；利用数据仓库前端的数据挖掘工具和人工智能技术，统计业务人员还可以建立各种统计调查、统计分析和统计预测模型，以分析国民经济、工农业产值、人口等领域的现状及发展变化趋势和方向。

利用数据仓库技术能够快速实现传统的统计报表、统计图形功能；更重要的是，利用数据仓库的数据挖掘技术可以使统计分析研究，无论从广度上还是深度上都有很大拓展，真正做到使其在统计预测和决策支持管理中发挥重要作用。

面对日新月异的信息技术，统计业面临以下三方面的需求。

(1) 数据的集中存储与管理

统计行业掌握着大量的、各历史年度的原始调查资料，受历史和技术(数据库存储处理能力的限制)等因素的制约，这些资料大都还保留在纸介质、脱机的磁带和软盘上。由于缺乏大型数据库的集中存储和统一管理，随着年代的增加，这些资料的保存和安全受到严峻的

考验;同时,这些宝贵的原始资料不能为统计业务人员随机查询和充分共享,不能进行有效的统计分析、预测评估和使用;难以快速地为管理决策提供科学依据。

(2) 查询方式和分析手段的更新

随着统计数据处理方式由逐级汇总到计算机超级汇总的转变,统计报表和统计分析需要从大量各种各样的原始材料中汇总整理各种不同需求、反映不同侧面的综合分析数据,传统的处理手段主要通过编写程序来实现,这样做的模式是固定的,且维护工作量大,开发周期长。为解决这种现状就需要一种技术或一种前端查询分析工具,统计业务人员可以根据任意条件、任意模式进行任意组合,生成查询结果,同时利用该工具能进行分析处理,能够方便地组成各种多维报表和统计图形,如条形图、饼图、曲线图、多维立方图等。另外,针对一些深层次的研究需要,还应提供一些统计分析智能软件和智能算法,以预测未来经济发展模式和走势。

(3) 与 Web 技术的有机结合

数据仓库技术与 Web 技术结合起来是采用目前流行的三层应用体系结构对系统进行的应用开发。所谓三层结构,是指后台是数据仓库,前台是 Web 服务器,客户端是浏览器的应用模式。利用这种技术,可以做到网上动态信息发布、网上随机查询和网上联机分析处理等功能,最终的目标是实现统计业务人员的日常工作完全在 Web 上实现。

针对以上需求,信息领域新技术的应用特别是数据仓库技术的应用是必然趋势。

2. 某市统计局企业微观数据仓库系统

实现某市统计局企业微观数据仓库是把掌握的不同专业、不同时期、分散的企业微观数据信息,按照多个主题集中存储和管理在数据仓库中,灵活地、非常方便地实现固定的和随机动态的数据查询处理、综合分析和统计报表。根据统计信息自动化总体规划要求,这些查询、分析和报表功能以及今后统计人员的日常业务处理工作都需在 Web 上进行。

在实现数据仓库之前,某市统计局已开发出企业微观数据库系统,受当时技术条件的限制,该系统的设计思路是按工业、建筑业、运输邮电业和批发零售贸易、餐饮业等不同专业分别建模,每个专业都对应一套数据存储表和管理字典,共性数据依照专业被分割、存储,这样做虽然数据管理条理清楚,安全性能好,查询方式易于接受,但存在的问题是查询方式不够灵活,不同专业的指标横向比较困难,难以实现产、供、销等企业生产各个阶段数据的一条龙分析研究。同时受软件条件限制,无法实现 Web 方式查询且速度较慢。数据仓库是面向主题建模,在进行设计的时候,将企业微观数据仓库设计成以下主题。

(1) 企业基本情况:各年度、各专业统计调查单位基本情况名录的主要内容及全部标识性内容。

(2) 企业财务状况:各年度、各专业企业的资产、经营投入、产出效益等财务经营状况。

(3) 企业劳动状况:各年度、各专业企业的就业人数及工资收入情况。

(4) 企业消耗状况:各年度、各专业企业生产所需的原材料及能源消耗情况,包括价值量和实物量消耗情况。

(5) 企业生产状况:各年度、各专业企业的主营生产情况。由于不同专业的生产方式不同,又下设若干子方面及工业产品产、销、存情况,建筑业生产完成情况,公路、水运、港口

企业生产完成情况，商业、餐饮业销售经营情况等。

这样建模以后，不同年度、不同专业的同类数据被集中进行存储，如此一来，指标无论是横向比较还是纵向比较都非常容易，并且整个系统只需要维护一套数据字典即可。

数据建模是数据仓库设计中非常重要的一个环节，包括逻辑建模和物理建模。在企业微观数据仓库中是利用 ERWIN 专业工具来建立模型，并形成相应的数据库结构。企业微观数据仓库的源数据是历年存储到微机上的数据，数据的格式、存储方式不尽相同，在加载到数据仓库之前，这些数据必须经过净化筛选、加工整理以及数据集成。利用 NCR 提供的 FastLoad 和其他工具，能方便地将经过处理的数据加载到 NCR 数据仓库里。目前企业微观数据仓库已存储 2 年各 4 个专业的历史数据，其他年度的数据正在整理当中。

应用开发的模式是基于目前流行的三层结构，即：后台是数据仓库，前台是 Web 服务器，客户端是浏览器。Brio Enterprise 商业智能工具提供了很好的基于 Web 浏览器的查询、联机分析及报表功能，并且具有极高的安全性和严格的权限访问等级。企业微观数据仓库系统的前端应用都是基于 Web 方式开发的，具有网上随机查询、网上多维分析、网上数据钻取、网上图形分析、网上表格旋转透视、网上多维报表等功能，并且操作方式都是拖拉方式，今后统计业务人员的月报、年报等数据处理都可以在网上进行。这样，数据仓库的好处、效益和威力发挥得淋漓尽致。

5.3.3 沃尔玛数据仓库系统

美国的沃尔玛（Wal-Mart）是世界上最大的零售商，2002 年 4 月，该公司跃居《财富》500 强企业排行第一。在全球拥有 4000 多家分店和连锁店。Wal-Mart 建立了基于 NCR Teradata 数据仓库的决策支持系统，它是世界上第二大数据仓库系统，总容量达到 170TB 以上。

沃尔玛成功的重要因素是与其充分地利用信息技术分不开的。也可以说，对信息技术的成功运用造就了沃尔玛。强大的数据仓库系统将世界 4000 多家分店的每一笔业务数据汇总到一起，让决策者能够在很短的时间里获得准确和及时的信息，并作出正确和有效的经营决策。而沃尔玛的员工也可以随时访问数据仓库，以获得所需的信息，而这并不会影响数据仓库的正常运转。关于这一点，沃尔玛的创始人萨姆·沃尔顿在他的自传《Made in America：My Story》一书是这样描述的："你知道，我总是喜欢尽快得到那些数据，我们越快得到那些信息，我们就能越快据此采取行动，这个系统已经成为我们的一个重要工具"。沃尔玛的数据仓库始建于 20 世纪 80 年代。自 1980 年以来，NCR 一直在帮助沃尔玛经营世界上最大的数据仓库系统。1988 年沃尔玛数据仓库容量为 12GB，1989 年升级为 24GB，以后逐年增长，1996 年其数据量达 7.5TB，1997 年为了圣诞节的市场预测和分析，沃尔玛将数据仓库容量扩展到 24TB。而到了信息技术飞速发展的今天，沃尔玛的数据仓库已经惊人地达到了超过 170TB。利用数据仓库，沃尔玛对商品进行市场类组分析（Marketing Basket Analysis），即分析哪些商品，顾客最有希望一起购买。沃尔玛数据仓库里集中了各个商店一年多详细的原始交易数据。在这些原始交易数据的基础上，沃尔玛利用自动数据挖掘工具（模式识别软件）对这些数据进行分析和挖掘。一个意外的发现就是：跟尿布一起购买最多的商品竟是啤酒！按常规思维，尿布与啤酒风马牛不相及，若不是借助于数据仓库系统，商家决不可能发现隐藏在背后的事实：原来美国的太太们常叮嘱她们的丈夫下班后

为小孩买尿布,而丈夫们在买尿布后又随手带回了两瓶啤酒。既然尿布与啤酒一起购买的机会最多,沃尔玛就在它的一个个商店里将它们并排摆放在一起,结果是尿布与啤酒的销量双双增长。由于这个故事的传奇和出人意料,所以一直被业界和商界所传诵。

这个故事仅仅是沃尔玛借助数据仓库受益的一连串成功故事的一个花絮而已。如今,沃尔玛利用 NCR 的 Teradata 对超过 7.5TB 的数据进行存储,这些数据主要包括各个商店前端设备(POS、扫描仪)采集来的原始销售数据和各个商店的库存数据。Teradata 数据库里存有 196 亿条记录,每天要处理并更新 2 亿条记录,要对来自 6000 多个用户的 48 000 条查询语句进行处理。销售数据、库存数据每天夜间从 4000 多个商店自动采集过来,并通过卫星线路传到总部的数据仓库里。沃尔玛数据仓库里最大的一张表格(table)容量已超过 300GB、存有 50 亿条记录,可容纳 65 个星期 4000 多个商店的销售数据,而每个商店有 5~8 万个商品品种。利用数据仓库,沃尔玛在商品分组布局、降低库存成本、了解销售全局、进行市场分析和趋势分析等方面进行决策支持分析,具体表现为以下几点。

1. 商品分组布局

作为微观销售的一种策略,合理的商品布局能节省顾客的购买时间,能刺激顾客的购买欲望。沃尔玛利用前面提到的市场类组分析(MBA),分析顾客的购买习惯,掌握不同商品一起购买的概率,甚至考虑购买者在商店里所穿行的路线、购买时间和地点,从而确定商品的最佳布局。

2. 降低库存成本

加快资金周转、降低库存成本是所有零售商面临的一个重要问题。沃尔玛通过数据仓库系统,将成千上万种商品的销售数据和库存数据集中起来,通过数据分析,以决定对各个商店各色货物进行增减,确保正确的库存。数十年来,沃尔玛的经营哲学是“代销”供应商的商品,也就是说,在顾客付款之前,供应商是不会拿到它的货款的。NCR 的 Teradata 数据仓库使他们的工作更具成效。数据仓库强大的决策支持系统每周要处理 25 000 个复杂查询,其中很大一部分来自供应商,库存信息和商品销售预测信息通过电子数据交换(EDI)直接送到供应商那里。数据仓库系统不仅使沃尔玛省去了商业中介,还把定期补充库存的担子转嫁到供应商身上。1996 年,沃尔玛开始通过 Web 站点销售商品,商品都是从供应商处直接订货。Web 站点销售相当成功,在其投入运营的第一个周末就卖出了一百多万件商品。

3. 了解销售全局

各个商店在传送数据之前,先对数据进行如下分组:商品种类、销售数量、商店地点、价格和日期等。通过这些分类信息,沃尔玛能对每个商店的情况有个细致的了解。在最后一家商店关门后一个半小时,沃尔玛已确切地知道当天的运营和财政情况。凭借对瞬间信息的随时捕捉,沃尔玛对销售的每一点增长,库存货物百分比的每点上升和通过削价而提高的每一份销售额都了如指掌。

4. 市场分析

沃尔玛利用数据挖掘工具和统计模型对数据仓库的数据仔细研究，以分析顾客的购买习惯、广告成功率和其他战略性的信息。沃尔玛每个星期六的高级会议上要对世界范围内销售量最大的15种商品进行分析，然后确保在准确的时间、合适的地点满足有所需要的库存。

5. 趋势分析

沃尔玛利用数据仓库对商品品种和库存的趋势进行分析，以选定需要补充的商品，研究顾客购买趋势，分析季节性购买模式，确定降价商品，并对其数量和运作作出反应。为了能够预测出季节性销售量，要检索数据仓库拥有100 000种商品一年多来的销售数据，并在此基础上作分析和知识挖掘。

萨姆·沃尔顿在他的自传中写道："我能顷刻之间把信息提取出来，而且是所有的数据。我能拿出我想要的任何东西，并确切地讲出我们卖了多少。"这感觉就像在信息的海洋里，"轻舟已过万重山"。他还写到："我想我们总是知道那些信息赋予你一定的力量，而我们能在计算机内取出这些数据的程度会使我们具有强大的竞争优势。"

沃尔玛神奇的增长很大部分可以归功于成功地建立了基于NCR Teradata的数据仓库系统。数据仓库改变了沃尔玛，而沃尔玛改变了零售业。在它的影响下，世界顶尖零售企业：Sears、Kmart、JCPenney、No. 1GermanRetailer、日本西武、三越等先后建立了数据仓库系统。沃尔玛的成功给人以启示：惟有站在信息巨人的肩头，才能掌握无限，创造辉煌。

习　题

1. 数据仓库管理与数据库管理有什么本质不同？
2. 数据仓库的数据为什么会增长？
3. 数据仓库管理主要包括哪些部分？
4. 信息使用者和探索者的任务有什么不同？
5. 信息使用者的性能需求是什么？
6. 有哪些方法来满足信息使用者的性能需求？
7. 为什么增加数据冗余能提高查询速度？
8. 什么是预聚集数据？
9. 什么是合并查询？
10. 探索者所做的工作有哪些？
11. 如何满足探索者的性能需求？
12. 什么是休眠数据？如何产生？
13. 删除休眠数据有哪些方法？
14. 什么是脏数据？如何产生？
15. 清理脏数据有哪些方法？

16. 监视休眠数据分哪三级？处理对策是什么？
17. 如何处理不同类型的脏数据？
18. 在数据仓库环境中元数据如何发挥作用？
19. 哪种元数据价值最大？
20. 元数据是不是知识？
21. 元数据的存储方法是什么？
22. 元数据交换标准(MDIS)的作用是什么？
23. 举例说明企业需要的战略信息。
24. 简述数据仓库查询服务内容。
25. 简述数据仓库报表服务内容。
26. 说明如何利用数据仓库找出出现问题的原因。
27. 说明如何利用数据仓库进行预测。
28. 数据仓库如何实现实时决策？
29. 数据仓库如何实现自动决策？
30. 对 5.2.2 小节中原因分析的实例,设计和画出决策支持系统结构图。
31. 在国内某市统计局数据仓库中选出两个主题画出星型模型图。
32. 利用沃尔玛数据仓库系统说明数据仓库的价值。

第 6 章　数据挖掘原理

6.1　知识发现过程

6.1.1　知识发现过程定义

知识发现被认为是从数据中发现有用知识的整个过程。数据挖掘被认为是 KDD 过程中的一个特定步骤，它用专门算法从数据中抽取模式(pattern)。

KDD 过程定义为(Fayyad、Piatetsky-Shapiror 和 Smyth，1996 年)：KDD 是从数据集中识别出有效的、新颖的、潜在有用的，以及最终可理解的模式的高级处理过程。

其中，数据集：事实 F(数据库元组)的集合。模式：用语言 L 表示的表达式 E，它所描述的数据是集合 F 的一个子集 F_E，它比枚举所有 F 中元素更简单，称 E 为模式。有效、新颖、潜在有用、可被人理解：表示发现的模式有一定的可信度，应该是新的，将来有实用价值，能被用户所理解。

KDD 过程如图 6.1 所示。

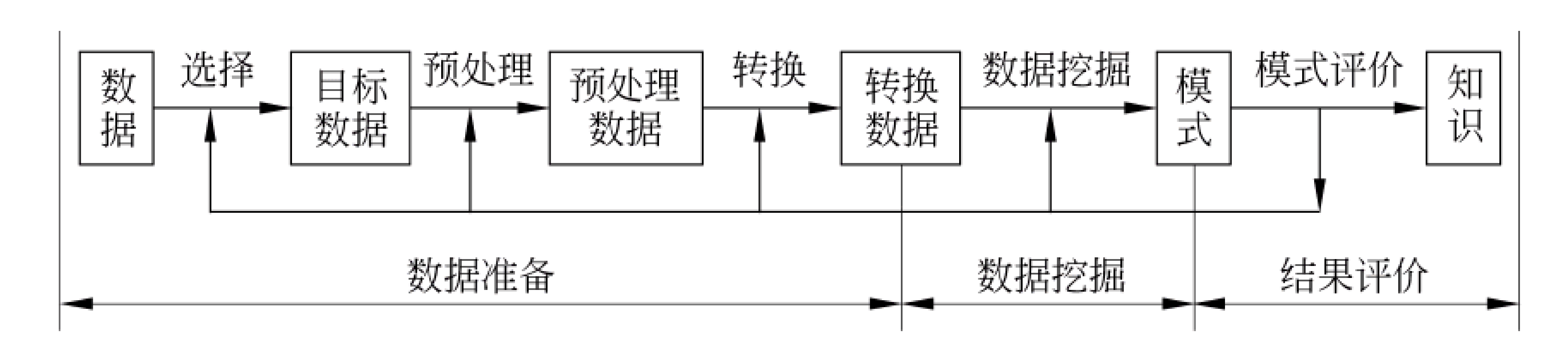


图 6.1　KDD 过程图

KDD 过程可以概括为 3 部分：数据准备(data preparation)、数据挖掘及结果的解释和评估(interpretation & evaluation)。

1. 数据准备

数据准备又可分为 3 个子步骤：数据选择(data selection)、数据预处理(data preprocessing)和数据转换(data transformation)。

数据选择的目的是确定发现任务的操作对象，即目标数据(target data)是根据用户的需要从原始数据库中选取的一组数据。数据预处理一般包括消除噪声、推导计算缺值数据、消除重复记录等。数据转换的主要目的是完成数据类型转换(如把连续值数据转换为离散型数据，以便于符号归纳，或是把离散型数据转换为连续值型数据，以便于神经网络计算)，尽量消减数据维数或降维(dimension reduction)，即从初始属性中找出真正有用的属性，以减少数据挖掘时要考虑的属性个数。

2. 数据挖掘

数据挖掘阶段首先要确定挖掘的任务或目的，如数据分类、聚类、关联规则发现或序列模式发现等。确定了挖掘任务后，就要决定使用什么样的挖掘算法。选择实现算法有两个考虑因素：一是不同的数据有不同的特点，因此需要用与之相关的算法来挖掘；二是用户或实际运行系统的要求，有的用户可能希望获取描述型的（descriptive）、容易理解的知识（采用规则表示的挖掘方法显然要好于神经网络之类的方法），而有的用户只是希望获取预测准确度尽可能高的预测型（predictive）知识。选择了挖掘算法后，就可以实施数据挖掘操作，获取有用的模式。

3. 结果的解释和评估

数据挖掘阶段发现出来的模式，经过评估，可能存在冗余或无关的模式，这时需要将其剔除；也有可能模式不满足用户要求，这时则需要回退到发现过程的前面阶段，如重新选取数据、采用新的数据变换方法、设定新的参数值，甚至换一种挖掘算法，等等。另外，KDD 由于最终是面向人类用户的，因此可能要对发现的模式进行可视化，或者把结果转换为用户易懂的另一种表示，如把分类决策树转换为 if…then…规则。

数据挖掘仅仅是整个过程中的一个步骤。数据挖掘质量的好坏有两个影响要素：一是所采用的数据挖掘技术的有效性，二是用于挖掘的数据的质量和数量（数据量的大小）。如果选择了错误的数据或不适当的属性，或对数据进行了不适当的转换，则挖掘的结果是不会好的。

整个挖掘过程是一个不断反馈的过程。比如，用户在挖掘途中发现选择的数据不太好，或使用的挖掘技术产生不了期望的结果，这时，用户需要重复先前的过程，甚至从头重新开始。

可视化技术在数据挖掘的各个阶段都扮演着重要的作用。特别是在数据准备阶段，用户可能要使用散点图、直方图等统计可视化技术来显示有关数据，以期对数据有一个初步的了解，从而为更好地选取数据打下基础。在挖掘阶段，用户则要使用与领域问题有关的可视化工具。在表示结果阶段，则可能要用到可视化技术，以使得发现的知识更易于理解。

6.1.2 数据挖掘对象

数据挖掘的对象主要是关系数据库和数据仓库，这是典型的结构化数据。随着技术的发展，数据挖掘对象逐步扩大到半结构化或非结构化数据，这主要是文本数据、图像与视频数据以及 Web 数据等。

1. 关系数据库

目前，建立的数据库都是关系数据库，数据仓库的数据存储仍然是关系数据库。数据挖掘方法也主要是研究数据库中属性之间的关系，挖掘出多个属性取值之间的规则。由于关系数据库的特点，促使了数据挖掘方法的改善。数据库的特点如下。

（1）数据动态性

数据的动态变化是数据库的一个主要特点。由于数据的存取和修改，使数据的内容经常发生变化，这就要求数据挖掘方法能适应这种变化。渐增式数据挖掘方法就是针对数据变化后，挖掘的规则知识能满足变化后的数据库内容。

(2) 数据不完全性

这主要反映在数据库中记录的域值丢失或不存在(空值)。这种不完全数据给数据挖掘带来了困难。为此，必须对数据进行预处理，填补该数据域的可能值。

(3) 数据噪声

由于数据录入等原因，造成错误的数据，即数据噪声。挖掘含噪声的数据会影响获取模式的准确性，并增加了数据挖掘的困难度。

在数据挖掘中要考虑噪声的影响，利用概率方法排除这些噪声。

(4) 数据冗余性

这表现在同一信息在多处重复出现。函数依赖是一个通常的冗余形式。冗余信息可能造成错误的数据挖掘，至少有些挖掘的知识是用户不感兴趣的。为避免这种情况的发生，数据挖掘时，需要知道数据库中有哪些固有的依赖关系。

(5) 数据稀疏性

这表现在多维数据空间中存在大量稀疏数据，稀疏数据会使数据挖掘丢失有用的模式。

(6) 海量数据

数据仓库中数据在不断增长，已出现很多海量数据仓库。数据挖掘方法需要逐步适应这种海量数据和迅速增长的数据挖掘，如建立有效的索引机制和快速查询方法等。

2. 文本

文本是以文字串形式表示的数据文件。文本分析包括：关键词或特征提取；相似检索；文本聚类和文本分类等。

(1) 关键词或特征提取

一篇文本中，标题是该文本的高度概括。标题中的关键词是标题的核心内容。关键词的提取对于掌握该文本的内容至关重要。

文本中的特征，如人名、地名、组织名等是某些文本中的主体信息，特征提取对掌握该文本的内容很重要。

(2) 相似检索

文本中的关键词的相似检索是了解文本内容的一种重要方法。例如“专家系统”与“人工智能”两个关键词是有一定联系的。研究专家系统的文本一定属于人工智能的研究领域。

(3) 文本聚类

对于文本标题中关键词(主题字)的相似匹配是对文本聚类的一种简单方法。定义关键词的相似度将便利文本的简单聚类，使类中的文本均满足关键词的相似度，使类间的文本的关键词一定超过相似度。

(4) 文本分类

将文本分类到各文本类中，一般需要采用一个算法。这些算法包括分类器算法、近邻算法等。这需要按文本中的关键词或特征的相似度来区分。

3. 图像与视频数据

图像与视频数据是典型多媒体数据。数据以点阵信息及帧形式存储,数据量很大。图像与视频的数据挖掘包括图像与视频特征提取、基于内容的相似检索、视频镜头的编辑与组织等。

(1) 图像与视频特征提取

图像与视频数据特征有颜色、纹理和形状等。这些特征提取用于基于内容的相似检索。海水蓝色、海滩黄色、房屋的形状及颜色,需要从大量图像和视频数据中提取。

(2) 基于内容的相似检索

根据图像、视频特征的分布、比例等进行基于内容的相似检索,可以将图像和视频数据进行聚类以及分类,也能完成对新图像或视频的识别,如对遥感图像或视频的识别。这种应用非常广泛,例如森林火灾的发现与报警,河流水灾的预报等。

(3) 视频镜头的编辑与组织

镜头代表一段连续动作(视频数据流)。典型的镜头编辑,如足球赛的射门、某段新闻节目等,需要在冗长的视频数据流中进行自动裁取。

经过编辑的镜头,按某种需要重新组织,将形成特定需求的新视频节目,如足球射门集锦、某个新闻事件的连续报道等。

4. Web 数据

随着 Internet 网的发展和普及,网站数目的迅速增长以及入网人员的急剧增多,使网络数据量呈指数增长。Web 数据挖掘已成为新课题。Web 数据挖掘的特点如下:

(1) 异构数据集成和挖掘

Web 上每一个站点是一个数据源,各数据源都是异构的,形成了一个巨大的异构数据库环境。将这些站点的异构数据进行集成,给用户提供一个统一的视图,才能在 Web 上进行数据挖掘。

(2) 半结构化数据模型抽取

Web 上的数据非常复杂,没有特定的模型描述。虽然每个站点上的数据是结构化的,但各自的设计对整个网络是一个非完全结构化的数据,称为半结构化数据。

对半结构化数据模型的查询和集成,需要寻找一种半结构化模型抽取技术来自动抽取各站点的数据。

XML 是一种半结构化的数据模型,容易实现 Web 中信息共享与交换。

采用"实时建议"技术,能够根据用户以往的浏览行为来预测该用户以后的浏览行为,从而为用户提供个性化的浏览建议。

总之,Web 数据挖掘正在逐步形成热点。

6.1.3 数据挖掘任务

数据挖掘任务有 6 项:关联分析、时序模式、聚类、分类、偏差检测、预测。

1. 关联分析

关联分析是从数据库中发现知识的一类重要方法。若两个或多个数据项的取值之间重复出现且概率很高时,就存在某种关联,可以建立起这些数据项的关联规则。

例如,买面包的顾客有 90%的人还买牛奶,这是一条关联规则。若商店中将面包和牛奶放在一起销售,将会提高销量。

在大型数据库中,这种关联规则是很多的,需要进行筛选,一般用“支持度”和“可信度”两个阈值来淘汰那些无用的关联规则。

“支持度”表示该规则所代表的事例(元组)占全部事例(元组)的百分比,如买面包又买牛奶的顾客占全部顾客的百分比。

“可信度”表示该规则所代表事例占满足前提条件事例的百分比。如买面包又买牛奶的顾客占买面包顾客中的 90%,称可信度为 90%。

2. 时序模式

通过时间序列搜索出重复发生概率较高的模式。这里强调时间序列的影响。例如,在所有购买了激光打印机的人中,半年后 80%的人再购买新硒鼓,20%的人用旧硒鼓装碳粉;在所有购买了彩色电视机的人中,有 60%的人再购买 VCD 产品。

在时序模式中,需要找出在某个最小时间内出现比率一直高于某一最小百分比(阈值)的规则。这些规则会随着形式的变化作适当的调整。

时序模式中,一个有重要影响的方法是“相似时序”。用“相似时序”的方法,要按时间顺序查看时间事件数据库,从中找出另一个或多个相似的时序事件。例如在零售市场上,找到另一个有相似销售的部门,在股市中找到有相似波动的股票。

3. 聚类

数据库中的数据可以划分为一系列有意义的子集,即类。简单地说,在设有类的数据中,按“距离”概念聚集成若干类。在同一类别中,个体之间的距离较小,而不同类别上的个体之间的距离偏大。聚类增强了人们对客观现实的认识,即通过聚类建立宏观概念。例如将鸡、鸭、鹅等都聚类为家禽。

聚类方法包括统计分析方法、机器学习方法、神经网络方法等。

在统计分析方法中,聚类分析是基于距离的聚类,如欧氏距离、海明距离等。这种聚类分析方法是一种基于全局比较的聚类,需要考察所有的个体才能决定类的划分。

在机器学习方法中,聚类是无导师的学习。在这里距离是根据概念的描述来确定的,故聚类也称概念聚类,当聚类对象动态增加时,概念聚类则称概念形成。

在神经网络中,自组织神经网络方法用于聚类。如 ART 模型、Kohonen 模型等,这是一种无监督学习方法。当给定距离阈值后,各样本按阈值进行聚类。

4. 分类

分类是数据挖掘中应用最多的任务。分类是在聚类的基础上对已确定的类找出该类别

的概念描述，代表了这类数据的整体信息，既该类的内涵描述，一般用规则或决策树模式表示。该模式能把数据库中的元组映射到给定类别中的某一个。

一个类的内涵描述分为特征描述和辨别性描述。

特征描述是对类中对象的共同特征的描述。辨别性描述是对两个或多个类之间的区别的描述。特征描述允许不同类中具有共同特征，而辨别性描述对不同类不能有相同特征。辨别性描述用的更多。

分类是利用训练样本集(已知数据库元组和类别所组成的样本)通过有关算法而求得。

建立分类决策树的典型方法有 ID3、C4.5、IBLE 等。建立分类规则的方法，典型的有 AQ 方法、粗集方法、遗传分类器等。

目前，分类方法的研究成果较多，判别方法的好坏可从 3 个方面进行：预测准确度(对非样本数据的判别准确度)；计算复杂度(方法实现时对时间和空间的复杂度)；模式的简洁度(在同样效果情况下希望决策树小或规则少)。

在数据库中往往存在噪声数据(错误数据)、缺损值、疏密不均匀等问题，它们对分类算法获取的知识将产生坏的影响。

5. 偏差检测

数据库中的数据存在很多异常情况，从数据分析中发现这些异常情况也是很重要的，以引起人们对它更多的注意。

偏差包括很多有用的知识，如：

(1) 分类中的反常实例；

(2) 模式的例外；

(3) 观察结果对模型预测的偏差；

(4) 量值随时间的变化。

偏差检测的基本方法是寻找观察结果与参照之间的差别。观察常常是某一个域的值或多个域值的汇总。参照是给定模型的预测、外界提供的标准或另一个观察。

6. 预测

预测是利用历史数据找出变化规律，建立模型，并用此模型来预测未来数据的种类、特征等。

典型的方法是回归分析，即利用大量的历史数据，以时间为变量，建立线性或非线性回归方程。预测时，只要输入任意的时间值，通过回归方程就可求出该时间的预测值。

近年来，发展起来的神经网络方法，如 BP 模型，实现了非线性样本的学习，能进行非线性函数的判别。

分类也能进行预测，但一般用于离散数值。回归预测用于连续数值。神经网络方法预测既可用于连续数值，也可以用于离散数值。

6.1.4 数据挖掘分类

数据挖掘涉及多个学科，主要包括数据库、统计学和机器学习三大主要技术。

数据库技术经过 20 世纪 80 年代的大发展，除关系数据库外，又陆续出现面向对象数据库、多媒体数据库、分布式数据库以及 Web 数据库等。数据库的应用由一般查询到模糊查询和智能查询，数据库计算已趋向并行计算。从以上各类数据库中挖掘知识正在兴起并已得到迅速发展。

统计学是门古老的学科，现已逐渐走向社会。它已成为社会调查、了解民意以及制定决策的重要手段。

机器学习是人工智能的重要分支。它是在专家系统获取知识出现瓶颈后发展起来的。机器学习的大部分方法和技术已成为数据挖掘方法和技术。

数据挖掘可按数据库类型、挖掘对象、挖掘任务、挖掘方法和技术，以及应用等几方面进行分类。

1. 按数据库类型分类

数据挖掘主要是在关系数据库中挖掘知识。随数据库类型的不断增加，逐步出现了不同数据库的数据挖掘。现有关系数据挖掘、模糊数据挖掘、历史数据挖掘、空间数据挖掘等多种不同数据库的数据挖掘类型。

2. 按数据挖掘对象分类

数据挖掘除对数据库这个主要对象进行挖掘外，还有文本数据挖掘、多媒体数据挖掘、Web 数据挖掘。由于对象不同，挖掘的方法相差很大，文本、多媒体、Web 数据均是非结构化数据，挖掘的难度将很大。

目前 Web 数据挖掘已逐步引起人们的关注。

3. 按数据挖掘任务分类

数据挖掘的任务有关联分析、时序模式、聚类、分类、偏差检测、预测等。按任务分类有：关联规则挖掘、序列模式挖掘、聚类数据挖掘、分类数据挖掘、偏差分析挖掘和预测数据挖掘等类型。

各类数据挖掘由于任务不同，将会采用不同的数据挖掘方法和技术。

4. 按数据挖掘方法和技术分类

数据挖掘方法和技术较多，在 6.2 节中详细讨论。在此对其分类进行说明。

(1) 归纳学习类

该类又分为基于信息论方法挖掘类和基于集合论方法挖掘类。基于信息论方法是在数据库中寻找信息量大的属性来建立属性的决策树。基于集合论方法是对数据库中各属性的元组集合之间关系(上、下近似关系，覆盖或排斥关系，包含关系等)来建立属性间的规则。各类中又包括多种方法，主要用于分类问题。

(2) 仿生物技术类

该类又分为神经网络方法类和遗传算法类。神经网络方法是在模拟人脑神经元而建立的 MP 数学模型和 Hebb 学习规则基础上，提出了一系列的算法模型，用于识别、预测、联

想、优化、聚类等实际问题。遗传算法是模拟生物遗传过程，对选择、交叉、变异过程建立了数学算子。主要用于问题的优化和规则的生成。

(3) 公式发现类

在科学实验与工程数据库中，用人工智能方法寻找和发现连续属性(变量)之间的关系，建立变量之间的公式，已引起人们的关注，该类中有多种数据挖掘方法，如 BACON 和 FDD 等。

(4) 统计分析类

统计分析是门独立学科，由于能对数据库中数据求出各种不同的统计信息和知识，故也构成了数据挖掘中一大类方法。

(5) 模糊数学类

模糊数学是反映人们思维的一种方式。将模糊数学应用于数据挖掘各项任务中，形成了模糊数据挖掘类，如模糊聚类、模糊分类、模糊关联规则等。

(6) 可视化技术类

可视化技术是一种图形显示技术。对数据的分布规律进行可视化显示或对数据挖掘过程进行可视化显示，会明显提高人们对数据挖掘的理解和挖掘效果。该技术已形成了可视化数据挖掘类的多种方法。

本书的内容将按数据挖掘的方法和技术分类的各种方法进行详细和深入的介绍，以便读者学习和使用这些方法和技术，对实际问题完成数据挖掘任务。

6.1.5 不完全数据处理

对不完全数据(incomplete data)的处理是知识发现过程中数据预处理的主要内容。在现实领域中，人们所拥有的数据常常是不完全的，在这种情况下，知识发现应该具有处理这种不完全数据并提供相应合理的近似结果的能力。

现实世界的数据库(例如商业数据库和医院数据库)中的数据很少是完全的：丢失的数据、观察不到的数据、隐藏的数据、录入过程中发生错误的数据等在现实中是经常发生的。在知识发现领域中对不完全数据的研究比较多的在于丢失的数据。

例如，在对个人调查时，被调查的对象可能会拒绝提供他的收入情况，在一项实验过程中，某些结果可能会因为某些故障而丢失，这些情况都会产生数据丢失。

关于两个变量 X 和 Y 的采样。其中 X 是独立变量，总有观测值；Y 是响应变量，可能涉及丢失值。以 $Y=?$ 代表丢失值，以 $(X=i, Y=?)$ 代表不完全的记录。由这种简单的两个变量模型，可以推广到更一般的情况，即一个不含丢失值的变量的集合总是影响着可能具有丢失值的另一个变量。这种情况在统计学、机器学习、数据挖掘和知识发现领域里是相当常见的。

丢失数据模式分类取决于 $Y=?$ 的概率是否依赖于 Y 与 X 的状态。如果这一概率 P 不依赖于 X 但依赖于 Y，则认为数据是随机丢失的(missing at random)；如果 $Y=?$ 的概率既不依赖于 Y 也不依赖于 X 的状态，则认为数据是完全随机丢失的(missing completely at random)。对于数据随机丢失和数据完全随机丢失两种情况，如果数据挖掘方法都不受影响，那么丢失数据的模式是可以忽略的。但当 $Y=?$ 的概率既依赖于 Y 又依赖于 X 时，则

丢失数据的模式就是不可忽略的。

处理丢失数据的方法如下。

1. 基于已知数据的方法

忽略掉丢失的数据而只对得到的数据进行挖掘和分析。这种方法最为简单，在数据量不太大且数据是完全随机丢失的情况下可以得到令人满意的结果。但是如果数据不是随机丢失的情况下，这种方法就不很有效，会导致严重的偏差，这时可以采用删除有丢失数据的属性方法。

2. 基于猜测的方法

首先猜测被丢失的值，从而得到完全的数据，然后再运用标准的统计学和机器学习的方法进行数据挖掘和分析。具体方法如下。

（1）均值替换法：用含有丢失值的属性的已知值的平均值来代替丢失的值。

（2）概率统计法：先求丢失值的所在属性的各取值的出现概率 $P(v_i^a)$，即表示属性 a 的取值 v_i 的出现概率。丢失值用出现最大概率的值 v 来代替。

（3）回归猜测：采用回归分析的方法，用未丢失的数据建立回归方程，用所依赖的变量 X 求出该丢失值 Y。

3. 基于模型的方法

对于丢失值构造出一个适当的模型（非回归模型），然后再在此模型下采用恰当的方法猜测丢失的值，这是一种较为灵活的方法。

4. 基于贝叶斯理论的方法

利用无教师指导的贝叶斯分类技术和贝叶斯网络处理丢失的数据。

5. 基于决策树的方法

利用决策树和规则归纳的技术来处理丢失的数据。

以上主要讨论了对不完全数据的处理。另外，对未知的数据、隐藏的数据、错误的数据等以及这些数据和已知数据的关系，目前研究较少，还需要深入研究。

6.1.6 数据库的数据浓缩

数据浓缩就是在满足某种等价条件下，将复杂的难以理解的数据库变换成简洁的、容易理解的高度浓缩的数据库。

数据浓缩包括两方面：属性约简和元组（记录）压缩。

1. 属性约简

属性约简一般用于分类问题。属性约简的原则是保持数据库中分类关系不变。目前，属性约简一般采用粗糙集（rough set）方法，也可以采用信息论方法。

在数据库(S)的分类问题中,属性分为条件属性(C)和决策属性(D)。属性约简是在条件属性中删除那些不影响对决策属性进行分类的多余的属性。经过研究对条件属性一般分为可省略属性和不可省略属性。不可省略属性实质上是对决策属性进行分类的核心属性(corse(S))。而可省略属性(choice(S))并不是全部都可省略的属性,需要在可省略属性中挑选出部分属性与核心属性组合成等价原数据库的分类效果。

例如,有如下汽车数据库(CTR),有 9 个条件属性和 1 个决策属性(里程),如表 6.1 所示。

表 6.1　汽车数据库(CTR)

序号	类型 a	汽缸 b	涡轮式 c	燃料 d	排气量 e	压缩率 f	功率 g	换挡 h	重量 i	里程 D
1	小型	6	Y	1 型	中	高	高	自动	中	中
2	小型	6	N	1 型	中	中	高	手动	中	中
3	小型	6	N	1 型	中	高	高	手动	中	中
4	小型	4	Y	1 型	中	高	高	手动	轻	高
5	小型	6	N	1 型	中	中	中	手动	中	中
6	小型	6	N	2 型	中	中	中	自动	重	低
7	小型	6	N	1 型	中	中	高	手动	重	低
8	微型	4	N	2 型	小	高	低	手动	轻	高
9	小型	4	N	2 型	小	高	低	手动	中	中
10	小型	4	N	2 型	小	高	中	自动	中	中
11	微型	4	N	1 型	小	高	低	手动	轻	高
12	微型	4	N	1 型	中	中	中	手动	中	高
13	小型	4	N	2 型	中	中	中	手动	中	中
14	微型	4	Y	1 型	小	高	高	手动	中	高
15	微型	4	N	2 型	小	中	低	手动	中	高
16	小型	4	Y	1 型	中	中	高	手动	中	中
17	小型	6	N	1 型	中	中	高	自动	中	中
18	小型	4	N	1 型	中	中	高	自动	中	中
19	微型	4	N	1 型	小	高	中	手动	中	高
20	小型	4	N	1 型	小	高	中	手动	中	高
21	小型	4	N	2 型	小	高	中	手动	中	中

经过分析,可以得到:

Corse(S)={燃料,重量},Choice(S)={类型、涡轮式、汽缸、排气量、压缩率、功率、换挡}

保持数据库(S)分类关系不变的 7 个属性约简如下:

(1) {类型,燃料,排气量,重量 }4 个属性;

(2) {燃料,排气量,压缩率,重量}4 个属性;

(3) {类型,汽缸,燃料,压缩率,重量}5 个属性;

(4) {类型,燃料,压缩率,功率,重量}5 个属性;

(5) {类型,汽缸,燃料,功率,重量}5 个属性;

(6) {汽缸,燃料,压缩率,功率,重量}5 个属性;

(7) {类型,汽缸,涡轮式,燃料,换挡,重量}6 个属性。

以上 7 种属性约简都等价于原数据库的 9 个属性的决策分类。其中最小属性约简是(1)和(2)用 4 个属性就可以代替数据库中 9 个属性。利用最小属性约简(2),经过进一步处理,可以得到原数据库的等价数据库,如表 6.2 所示。

表 6.2　约简后的数据库

项目	燃料	排气量	压缩率	重量	里程
1’	*	*	*	重	低
2’	*	*	*	轻	高
3’	*	小	中	*	高
4’	*	中	*	中	中
5’	1 型	小	高	*	高
6’	2 型	*	高	中	中

表 6.2 中的 * 表示可不考虑该属性的取值。

2. 元组(记录)压缩

元组(记录)压缩实质上是对数据库的元组(记录)进行合并、归并和聚类等。

(1) 相同元组(记录)的合并

在进行属性约简后,会出现很多相同的元组。这样,可以合并这些相同的元组。

(2) 利用概念树进行归并

概念树是一种对概念的层次划分的树。概念树与数据库中特定的属性有关,它将各个层次的概念按一般到特殊的顺序排列。在概念树中最一般的概念作为树的根结点,最特殊的概念作为叶结点,它对应数据库具体属性值。例如,反映某数据库中“籍贯”这个属性的概念树如图 6.2 所示。

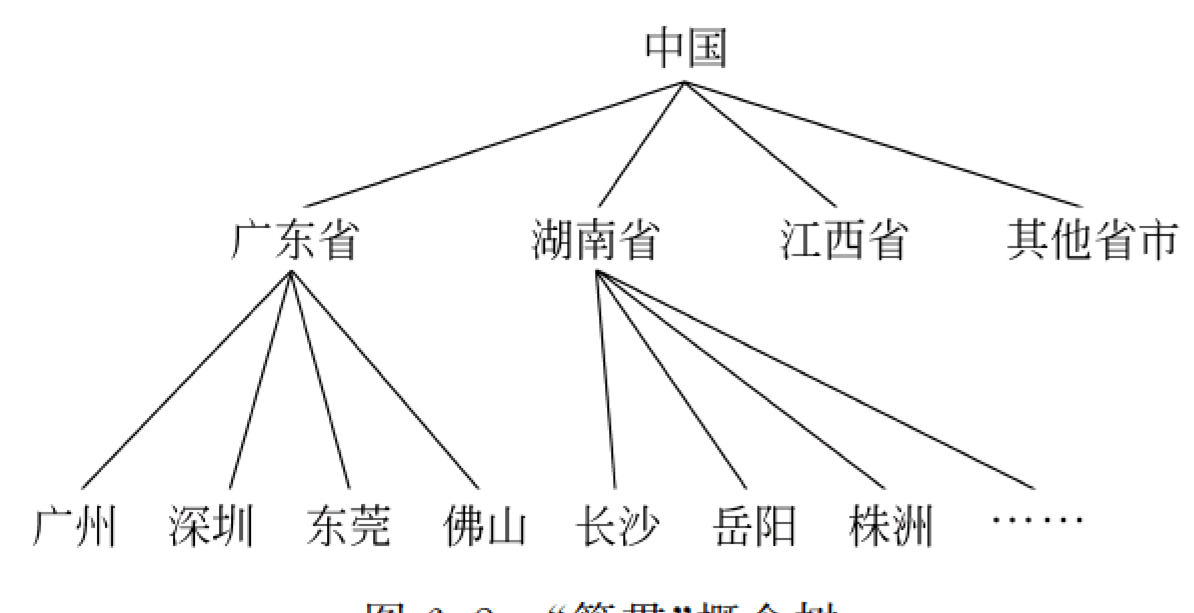


图 6.2　“籍贯”概念树

利用概念树进行向上归纳,可以实现数据库元组归并。例如,对数据库中“籍贯”为广州、深圳、东莞、佛山等城市的所有学生的记录都归并为广东省,即“籍贯=广东省”的新记录中,这样就完成了广东省内学生的多个元组(记录)都归并到一个元组(记录)中。实现了元组(记录)的压缩。对学生数据库这种元组压缩便于学校对各省学生的生活习惯有概括的了解,便于学校对学生的管理。

(3) 对元组的聚类

为了对数据库中所有元组(记录)有一个概括的了解,在元组之间设定一种距离方法(如海明距离),对数据库中所有元组进行聚类。这种聚类能完成对同一类的多个元组进行聚集,形成一个类元组。数据库按类元组重新组织,就完成了原数据库元组高度压缩的新数据库。

6.2 数据挖掘方法和技术

数据挖掘方法是由人工智能、机器学习的方法发展而来,结合传统的统计分析方法、模糊数学方法以及科学计算可视化技术,以数据库为研究对象,形成了数据挖掘方法和技术。

数据挖掘方法和技术可以分为 6 大类。

6.2.1 归纳学习的信息论方法

归纳学习方法是目前重点研究的方向,研究成果较多。从采用的技术上看,分为两大类:信息论方法(这也是常说的决策树方法)和集合论方法。每类方法又包含多个具体方法。

信息论方法是利用信息论的原理建立决策树。由于该方法最后获得的知识表示形式是决策树,故一般文献中称它为决策树方法。该类方法的实用效果好,影响较大。

信息论方法中较有特色的方法有:

1. ID3 等方法(决策树方法)

Quiulan 研制的 ID3 方法是利用信息论中互信息(Quiulan 称为信息增益)寻找数据库中具有最大信息量的字段,建立决策树的一个结点,再根据字段的不同取值建立树的分支,再由每个分支的数据子集重复建树的下层结点和分支的过程,这样就建立了决策树。这种方法对数据库愈大效果愈好。ID3 方法在国际上影响很大。继 ID3 方法以后陆续开发了 ID4、ID5、C4.5 等方法。

2. IBLE 方法(决策规则树方法)

钟鸣等人研制了 IBLE 方法,是利用信息论中信道容量寻找数据库中信息量从大到小的多个字段的取值,建立决策规则树的一个结点,根据该结点中指定字段取值的权值之和与两个阈值比较,建立左、中、右 3 个分支,在各分支子集中重复建树结点和分支的过程,就建立了决策规则树。IBLE 方法比 ID3 方法在识别率上提高了 10%。

6.2.2 归纳学习的集合论方法

集合论方法是开展较早的方法。近年来,由于粗糙集理论的发展使集合论方法得到了迅速的发展。这类方法中包括覆盖正例排斥反例的方法(典型的方法是 AQ 系列方法)、概念树方法和粗糙集(rough set)方法。关联规则挖掘也属于集合论方法。

1. 粗糙集(rough set)方法

在数据库中将行元素看成对象,列元素是属性(分为条件属性和决策属性)。等价关系 R 定义为不同对象在某个(或几个)属性上取值相同,这些满足等价关系的对象组成的集合称为该等价关系 R 的等价类。条件属性上的等价类 E 与决策属性上的等价类 Y 之间有 3 种情况:①下近似:Y 包含 E;②上近似:Y 和 E 的交非空;③无关:Y 和 E 的交为空。对下近似建立确定性规则,对上近似建立不确定性规则(含可信度),无关情况不存在规则。

2. 关联规则挖掘

关联规则挖掘是在交易事务数据库中挖掘出不同项(商品)集的关联关系,即发现哪些商品频繁地被顾客同时购买。

关联规则挖掘是在事务数据库 D 中寻找那些不同项集(如含 A 和 B 两个商品)同时出现的概率(即 $P(AB)$)大于最小支持度(min_sup),且在包含一个项集(如 A)的所有事务中,又包含另一个项集(如 B)的条件概率(即 $P(B|A)$)大于最小可信度(min_conf)时,则存在关联规则(即 $A \rightarrow B$)。

3. 覆盖正例排斥反例方法

它是利用覆盖所有正例、排斥所有反例的思想来寻找规则。比较典型的有 Michalski 的 AQ11 方法、洪家荣改进的 AQ15 方法以及洪家荣的 AE5 方法。

AQ 系列的核心算法是在正例集中任选一个种子,到反例集中逐个比较,对字段取值构成的选择子相容则舍去,相斥则保留。按此思想循环所有正例种子,将得到正例集的规则(选择子的合取式)。

AE 系列方法是在扩张矩阵中寻找覆盖正例排斥反例的字段值的公共路(规则)。

4. 概念树方法

数据库中记录的属性字段按归类方式进行合并,建立起来的层次结构称为概念树。如对"城市"概念树的最下层是具体市名或县名(如长沙、南京等),它的直接上层是省名(湖南、江苏等),省名的直接上层是国家行政区(华南、华东等),再上层是国名(中国、日本等)。

利用概念树提升的方法可以大大浓缩数据库中的记录(元组)。对多个属性字段的概念树提升,将得到高度概括的知识基表,再将它转换成规则。

6.2.3 仿生物技术的神经网络方法

仿生物技术典型的方法是神经网络方法和遗传算法。这两类方法已经形成了独立的研究体系。它们在数据挖掘中也发挥了巨大的作用,将它们归并为仿生物技术类。

神经网络方法是模拟了人脑神经元结构,以 MP 模型和 Hebb 学习规则为基础,建立三大类多种神经网络模型。

1. 前馈式网络

它以感知机、BP 反向传播模型、函数型网络为代表。此类网络可用于预测、模式识别等方面。

2. 反馈式网络

它以 Hopfield 的离散模型和连续模型为代表，分别用于联想记忆和优化计算。

3. 自组织网络

它以 ART 模型、Kohonen 模型为代表。它们用于聚类。

神经网络的知识体现在网络连接的权值上，是一个分布式矩阵结构。神经网络的学习体现在神经网络权值的逐步计算上(包括反复迭代或者是累加计算)。

6.2.4 仿生物技术的遗传算法

这是模拟生物进化过程的算法。它由 3 个基本算子组成：

1. 繁殖(选择)

从一个旧种群(父代)选择出生命力强的个体产生新种群(后代)的过程。

2. 交叉(重组)

选择两个不同个体(染色体)的部分(基因)进行交换，形成两个新个体。

3. 变异(突变)

对某些个体的某些基因进行变异(1 变 0，0 变 1)，形成新个体。

这种遗传算法起到产生优良后代的作用。这些后代需要满足适应值，经过若干代的遗传，将得到满足要求的后代(问题的解)。遗传算法已在优化计算和分类机器学习方面发挥了显著的效果。

6.2.5 数值数据的公式发现

在工程和科学数据库(由实验数据组成)中对若干数据项(变量)进行一定的数学运算，求得相应的数学公式。

1. 物理定律发现系统 BACON

BACON 发现系统完成了物理学中大量定律的重新发现。它的基本思想是对数据项进行初等数学运算(加、减、乘、除等)形成组合数据项，若它的值为常数项，就得到了组合数据项等于常数的公式，该系统有 5 个版本，分别为 BACON.1～BACON.5。

2. 经验公式发现系统 FDD

作者等人研制了 FDD 发现系统。基本思想是对两个数据项交替取初等函数后与另一

数据项的线性组合若为直线时，就找到了数据项(变量)的初等函数的线性组合公式。该系统所发现的公式比 BACON 系统发现的公式更宽些，该系统有 3 个版本，分别为 FDD.1～FDD.3。

6.2.6 可视化技术

可视化技术是一种图形显示技术。例如，把数据库中多维数据变成多种图形，这对于揭示数据中内在本质以及分布规律起到很强的作用。对数据挖掘过程可视化，并进行人机交互可提高数据挖掘的效果。

数据可视化是创建二维或三维业务数据集的图表，使得用户用于理解业务数据，从而提升知识和洞察力。例如，多维数据的多维结构类型(MTS)图与多维表格是对多维数据可视化的显示。利用直方图(二维)、柱形图(三维)、饼图、折线图、雷达图、散点图等能更形象地表示数据之间对比与变化的关系。

可视化数据挖掘是创建可视化的数据挖掘模型，利用这些模型发现业务数据集中存在的模式，从而辅助决策支持及预测新的商机。

可视化技术的基本工作为：

1. 提取几何图元

这是可视化系统的主要部分，由不同类型的数据(点、线)构造成表面或体素模型。它是构造、仿真、分析数据分布模型的有效手段。

2. 绘制

这是利用计算机图形学中的成果，进行图像生成、消隐、光照效应及绘制的部件。

3. 显示和演放

为了取得有效的显示效果，这一部件将提供图片组合、文件标准、着色、旋转、放大、存储等功能。

可视化绘制(render)方法就是把隐藏于大容量数据集中的物理信息转化为有组织结构表示的视觉信号集合，如空间几何形状、颜色、亮度等。目前常用的可视化绘制方法有几何法、彩色法、多媒体法和光学法。

6.3 数据挖掘的知识表示

数据挖掘的各种方法获得的知识表示形式主要有 6 种：规则、决策树、知识基(浓缩数据)、网络权值、公式和案例。

6.3.1 规则知识

规则知识由前提条件和结论两部分组成。前提条件由字段项(属性)的取值的合取(与∧)和析取(或∨)组合而成，结论为决策字段项(属性)的取值或者类别组成。

用一个简单例子进行说明,如两类人数据库的 9 个元组(记录)如表 6.3 所示。

表 6.3　两类人数据库

类别	身高	头发	眼睛
第一类人	矮	金色	蓝色
	高	红色	蓝色
	高	金色	蓝色
	矮	金色	灰色
第二类人	高	金色	黑色
	矮	黑色	蓝色
	高	黑色	蓝色
	高	黑色	灰色
	矮	金色	黑色

利用上面介绍的数据挖掘方法能很快得到如下规则知识:

IF(发色 = 金色 ∨ 红色) ∧ (眼睛 = 蓝色 ∨ 灰色)THEN 第一类人

IF(发色 = 黑色) ∨ (眼睛 = 黑色)THEN 第二类人

即:凡是具有金色或红色的头发,并且同时具有蓝色或灰色眼睛的人属于第一类人;凡是具有黑色头发或黑色眼睛的人属于第二类人。

6.3.2　决策树知识

数据挖掘的信息论方法所获得的知识一般表示为决策树。

如 ID3 方法的决策树是由信息量最大的字段(属性)作为根结点,它的各个取值为分支,对各个分支所划分的数据元组(记录)子集,重复建树过程,扩展决策树,最后得到相同类别的子集,以该类别作为叶结点。

例如,上例的两类人数据库,按 ID3 方法得到的决策树,如图 6.3 所示。

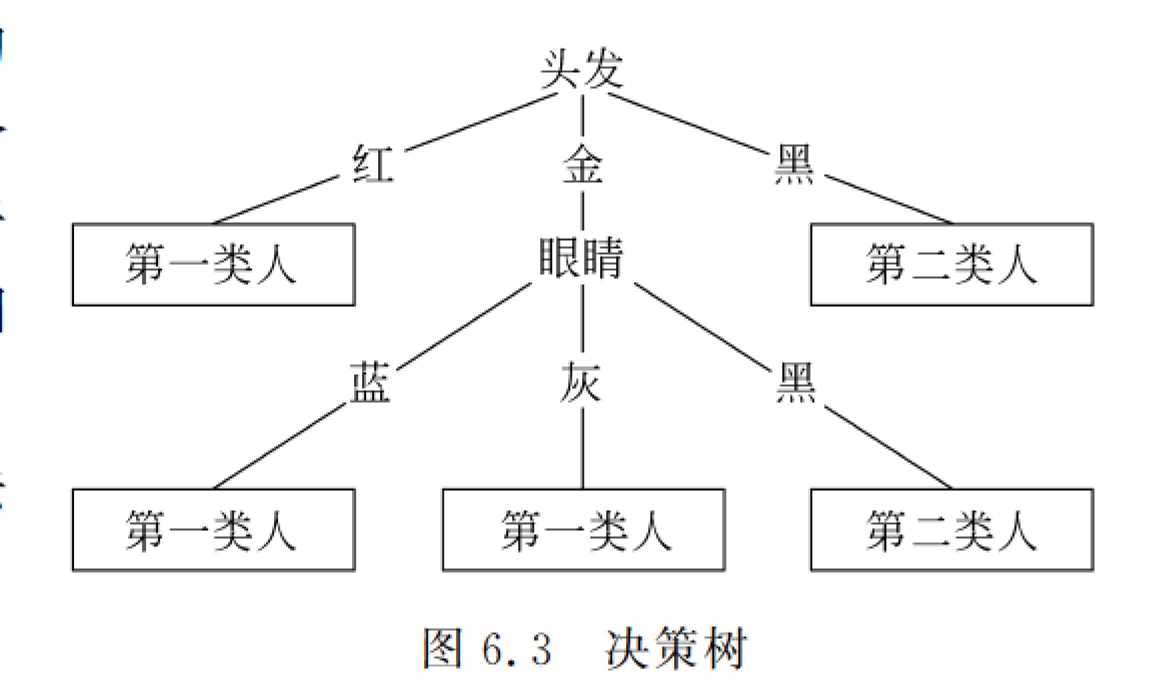


图 6.3　决策树

6.3.3　知识基

在知识发现过程的数据准备中,数据转换的一项属性约简工作是,找出可省略的属性。在删除不必要的属性后,对数据库中出现相同的元组(记录)进行合并。这样,通过属性约简方法能压缩数据库的属性和相应的元组,最后得到浓缩数据,称为知识基。它是原数据库的精华,很容易转换成规则知识。

例如上例的两类人数据库,通过属性约简计算可以得出身高是不必要的属性,删除它后,再合并相同数据元组,得到浓缩数据如表 6.4 所示。

表 6.4　知识基(浓缩数据)

类别	头发	眼睛
一类人	金色	蓝色
一类人	红色	蓝色
一类人	金色	灰色
二类人	金色	黑色
二类人	黑色	蓝色
二类人	黑色	灰色

6.3.4　神经网络的权值

神经网络方法经过对训练样本的学习后,所得到的知识是网络连接权值和结点的阈值。一般表示为矩阵和向量。例如,异或问题的网络权值和阈值如图 6.4 所示。

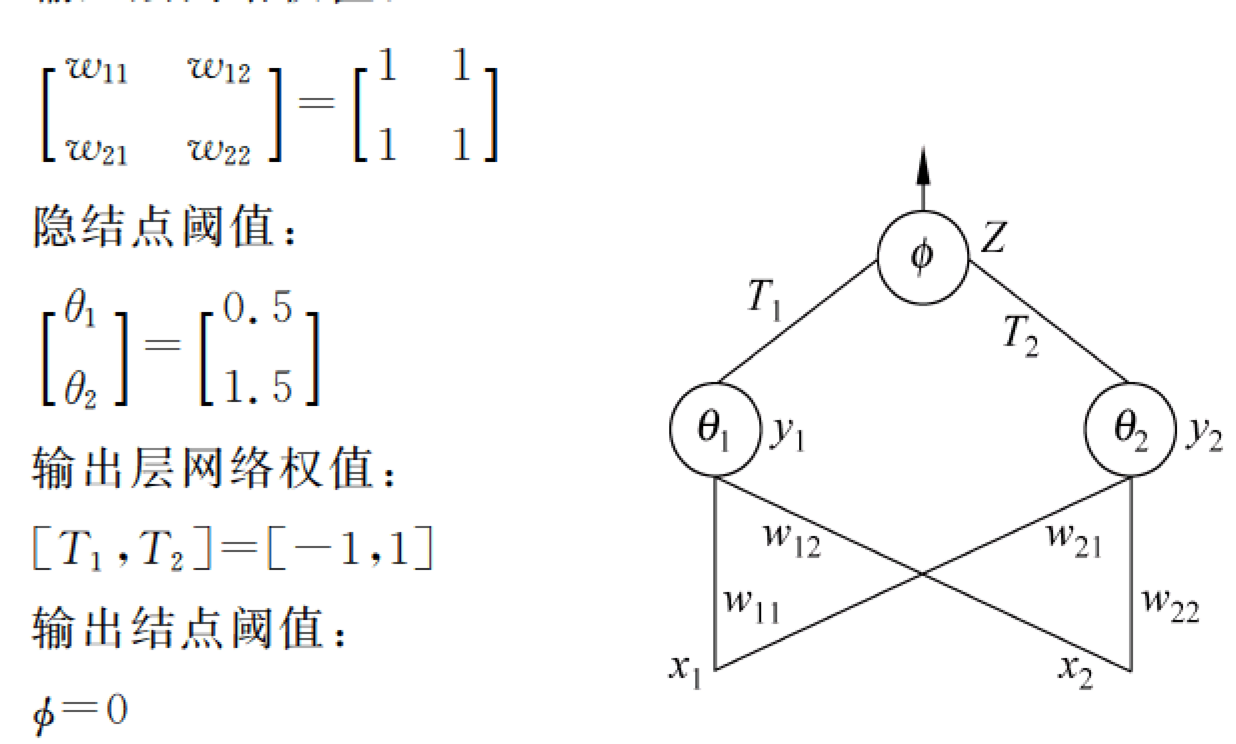


图 6.4　神经网络结构和权值

6.3.5　公式知识

对于科学和工程数据库,一般存放的是大量实验数据(数值)。它们中蕴涵着一定的规律性,通过公式发现算法,可以找出各种变量间的相互关系,并用公式表示。

例如,太阳系行星运动数据中包含行星运动周期(旋转一周所需时间,如天),以及它与太阳的距离(围绕太阳旋转的椭圆轨道的长半轴,如百万公里),数据如表 6.5 所示。

表 6.5　太阳系行星数据

类别	水星	金星	地球	火星	木星	土星
周期 p/天	88	225	365	687	4343.5	10 767.5
距离 d/100 万 km	58	108	149	228	778	1430

通过物理定律发现系统 BACON 和我们研制的经验公式发现系统 FDD 均可以得到开普勒第三定律:

$$d^3/p^2 = 25$$

6.3.6 案例

案例是人们经历过的一次完整的事件。当人们为解决一个新问题时,总是先回顾自己以前处理过的类似事件(案例)。利用以前案例中解决问题的方法或者处理的结果,作为参考并进行适当的修改,以解决当前新问题。利用这种思想建立起基于案例推理(case based reasoning,CBR)。CBR 的基础是案例库,在案例库中存放大量的成功或失败的案例。CBR 利用相似检索技术,对新问题到案例库中搜索相似案例,再经过对旧案例的修改来解决新问题。

可见,案例是解决新问题的一种知识。案例知识一般表示为三元组:

＜问题描述,解描述,效果描述＞

- 问题描述:待求解问题及周围世界或环境的所有特征的描述;
- 解描述:对问题求解方案的描述;
- 效果描述:描述解决方案后的结果情况,是失败还是成功。

习　　题

1. 知识发现过程由哪三部分组成?每部分的工作是什么?
2. 数据挖掘的对象有哪些?它们各自的特点是什么?
3. 数据挖掘的任务有哪些?每项任务的含义是什么?
4. 聚类与分类有什么不同?
5. 如何出现不完全数据?
6. 数据是随机丢失的概念是什么?
7. 数据是完全随机丢失的概念是什么?
8. 哪种丢失数据的模式是可以忽略的?
9. 哪种丢失数据的模式是不可以忽略的?
10. 处理丢失数据的方法有哪些?
11. 数据浓缩包括哪两个方面?
12. 属性约简的原则是什么?
13. 属性约简一般采用哪些方法?
14. 元组压缩有哪几种?
15. 利用概念树如何进行元组的压缩?
16. ID3 方法建立决策树的基本思想是什么?
17. "信息增益"是"互信息"吗?
18. 粗糙集方法如何获得规则?
19. 神经网络方法有哪几类?
20. 遗传算法的 3 个算子是什么?

21. 公式发现中的 BACON 方法与 FDD 方法的基本思想是什么？
22. 数据挖掘的知识表示有哪些？
23. 规则知识与决策树知识和知识基是等价的吗？
24. 人类社会的知识表示是什么？它与计算机中的知识表示有什么不同？
25. 为什么要研究计算机中的知识表示？

第7章 信息论方法

7.1 信息论原理

信息论是 C. E. Shannon 为解决信息传递(通信)过程问题而建立的理论,也称为统计通信理论。一个传递信息的系统是由发送端(信源)和接收端(信宿)以及连接两者的通道(信道)三者组成。信息论把通信过程看做是在随机干扰的环境中传递信息的过程。在这个通信模型中,信息源和干扰(噪声)都被理解为某种随机过程或随机序列。因此,在进行实际的通信之前,收信者(信宿)不可能确切了解信源究竟会发出什么样的具体信息,不可能判断信源会处于什么样的状态。这种情形就称为信宿对于信源状态具有不确定性。而且这种不确定性是存在于通信之前的。因而又叫做先验不确定性。

在进行了通信之后,信宿收到了信源发来的信息,这种先验不确定性才会被消除或者被减少。如果干扰很小,不会对传递的信息产生任何可察觉的影响,信源发出的信息能够被信宿全部收到,在这种情况下,信宿的先验不确定性就会被完全消除。但是,在一般情况下,干扰总会对信源发出的信息造成某种破坏,使信宿收到的信息不完全。因此,先验不确定性不能全部被消除,只能部分地消除。换句话说,通信结束之后,信宿还仍然具有一定程度的不确定性,这就是后验不确定性。显然,后验不确定性总要小于先验不确定性,不可能大于先验不确定性。

(1) 如果后验不确定性的大小正好等于先验不确定性的大小,就表示信宿根本没有收到信息。

(2) 如果后验不确定性的大小等于零,就表示信宿收到了全部信息。

可见,信息是用来消除(随机)不确定性的度量。信息量的大小由所消除的不确定性的大小来计量。

7.1.1 信道模型和学习信道模型

1. 信道模型

信息论的信道模型如图 7.1 所示。信源发出的符号 U 取值为 $u_1, u_2, \cdots, u_r$,信宿接收的符号 V 取值为 $v_1, v_2, \cdots, v_q$。

条件概率 $P(V|U)$ 称为信道的传输概率或转移概率,反映信道的输入与输出的关系,用矩阵来表示称为转移概率矩阵。

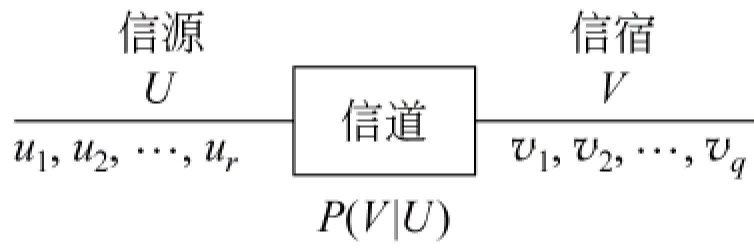


图 7.1 信道模型

$$\begin{bmatrix} P(v_1/u_1) & P(v_2/u_1) & \cdots & P(v_q/u_1) \\ P(v_1/u_2) & P(v_2/u_2) & \cdots & P(v_q/u_2) \\ \cdots & \cdots & \cdots & \cdots \\ P(v_1/u_r) & P(v_2/u_r) & \cdots & P(v_q/u_r) \end{bmatrix} \tag{7.1}$$

其中，$\sum_{i=1}^{r} P(v_j/u_i) = 1,\ j=1,2,\cdots,q$。

转移概率 $P(v_j/u_i)$表示收到信息 v_j 后判定输入为 u_i 的概率。

信道的数学模型可用三元组($U,P(V|U),V$)来表示，给定三元组后信道就给定了。给定了信道，将要研究在信宿收到符号 V 的值 v_j 后，如何正确判定信源发出的符号 U 是哪个值 u_i。

2. 学习信道模型

学习信道模型是信息模型应用于机器学习和数据挖掘的具体化。学习信道模型的信源是实体的类别，简单采用“是”、“非”两类，令实体类别 U 的值域为$\{u_1,u_2\}$，U 取 u_1 表示取“是”类中任一例子，取 u_2 表示取“非”类中任一例子。信宿是实体的特征(属性)取值。实体中某个特征(属性)V，它的值域为$\{v_1,v_2,\cdots,v_q\}$。

把实体中的类别 U 看成输入，把某特征的取值 V 看成输出，建立“学习信道模型”，如图 7.2 所示。

建立学习信道模型后，就可以利用信息论的信道模型原理来解决归纳学习和数据挖掘的问题。

U —— 学习信道 —— V

u_1, u_2 $P(V|U)$ $v_1, v_2, \cdots, v_q$

图 7.2　学习信道模型

7.1.2　信息熵和条件熵

1. 消息(符号)

$u_i(i=1,2,\cdots,r)$的发生概率 $P(u_i)$组成信源数学模型(样本空间和概率空间)

$$[U,P] = \begin{bmatrix} u_1 & u_2 & \cdots & u_r \\ P(u_1) & P(u_2) & \cdots & P(u_r) \end{bmatrix} \tag{7.2}$$

2. 自信息

消息 u_i 发生后所含有的信息量反映了消息 u_i 发生前的不确定性(随机性)，定义为

$$I(u_i) = \log \frac{1}{P(u_i)} = -\log P(u_i) \tag{7.3}$$

log 以 2 为底，所得的信息量单位为 b。

3. 信息熵

自信息的数学期望，即信源发出消息后，信源消息所提供的信息量，也反映了信源发出消息前的平均不确定性。定义为

$$H(U) = \sum_i P(u_i)\log \frac{1}{P(u_i)} = -\sum_i P(u_i)\log P(u_i) \tag{7.4}$$

信息熵 $H(U)$是信源发出前的平均不确定性，也称先验熵。$H(U)$的性质：

(1) $H(U)=0$ 时，说明只存在着惟一的可能性，不存在不确定性。

(2) 如果 n 种可能的发生都有相同的概率，即所有的 u_i 有 $P(u_i)=1/n$，$H(U)$达到最大值 $\log n$，系统的不确定性最大。

(3) $P(u_i)$互相接近，$H(U)$就大。$P(u_i)$相差大，则 $H(U)$就小。

如果信道中无干扰(噪声)，信道输出符号 v_j 与输入符号 u_i 一一对应，那么接收到传送过来的符号后就消除了对发送符号的先验不确定性。

4. 后验熵

一般信道中有干扰存在，信宿接收到符号 V 后对信源发出的是什么符号仍有不确定性。那么，怎样来度量接收到 V 后关于 U 的不确定性呢？当没有接收到输出符号 V 时，已知发出符号 U 的概率分布为 $P(U)$，而当接收到输出符号 $V=v_j$ 后，输入符号的概率分布发生了变化，变成后验概率分布 $P(U|v_j)$。那么接收到输出符号 $V=v_j$ 后，关于 U 的平均不确定性为

$$H(U \mid v_j) = \sum_i P(u_i \mid v_j) \log \frac{1}{P(u_i \mid v_j)} \tag{7.5}$$

这是接收到输出符号 v_j 后关于 U 的后验熵。后验熵是当信道接收端接收到输出符号 v_j 后，关于输入符号 U 的信息度量。

5. 条件熵

后验熵在输出符号集 V 的范围内是个随机量，对后验熵在输出符号集 V 中求期望，得到条件熵

$$H(U \mid V) = \sum_j P(v_j) \sum_i P(u_i \mid v_j) \log \frac{1}{P(u_i \mid v_j)} \tag{7.6}$$

这个条件熵称为信道疑义度。它表示在输出端收到全部输出符号 V 后，对于输入端的符号集 U 尚存在的不确定性(存在疑义)。对 U 集尚存在的不确定性是由于干扰(噪声)引起的。如果是一一对应信道，那么接收到符号集 V 后，对 U 集的不确定性完全消除，则信道疑义度 $H(U|V)=0$。

从上面分析可知：条件熵小于无条件熵，即 $H(U|V)<H(U)$。说明接收到符号集 V 的所有符号后，关于输入符号 U 的平均不确定性减少了，即总能消除一些关于输入端 U 的不确定性，从而获得了一些信息。

7.1.3 互信息与信息增益

$H(U)$代表接收到输出符号集 V 以前关于输入符号集 U 的平均不确定性，而 $H(U|V)$代表收到输出符号集 V 后关于输入符号 U 的平均不确定性。可见，通过信道传输消除了一些不确定性，获得了一定的信息。定义：

$$I(U,V) = H(U) - H(U \mid V) \tag{7.7}$$

$I(U,V)$称为 U 和 V 之间的互信息。它代表接收到符号集 V 后获得的关于 U 的信

息量。

可见，熵（$H(U)$、$H(U|V)$）只是平均不确定性的描述。熵差（$H(U)-H(U|V)$）是不确定性的消除，即互信息才是接收端所获得的信息量。

对输入端 U 只有 u_1、u_2 两类，互信息的计算公式为：

$$H(U)=\sum_{i=1}^{2}P(u_i)\log\frac{1}{P(u_i)}$$

$$H(U\mid V)=\sum_{j}P(v_j)\sum_{i}P(u_i\mid v_j)\log\frac{1}{P(u_i\mid v_j)}$$

$$I(U,V)=H(U)-H(U\mid V)$$

当 $P(u_i)$ 或 $P(u_i|v_j)$ 为零时，定义对数为零。

J. R. Quinlan 在提出 ID3 方法时，用“信息增益”概念，实际上是信息论中的“互信息”概念。

7.1.4 信道容量与译码准则

1. 信道容量

给定信道的互信息 $I(U,V)$ 是 $P(U)$ 的 ∩ 型函数。由 ∩ 型函数的性质知道，一定存在一概率分布 $P(U)$，使得 $I(U,V)$ 达到最大。这个最大的互信息就称为信道容量(capacity)，记为 C。

$$C=\max_{P(U)}\{I(U,V)\} \tag{7.8}$$

无论 $P(U)$ 如何变化，$I(U,V)$ 总不会大于 C。因此 C 对给定信道是个常数。

若以 C 作为特征选择量，去掉 C 小的特征（信息量小的特征），选择 C 大的特征（信息量大的特征），即 C 大的特征对区分正反例有效。

互信息 $I(U,V)$ 的计算会随实例个数的变化而变化，而信道容量 C 不会随实体个数的多少而变化，用 C 作为特征的信息量更准确。但是，C 的计算极为复杂，一般要用计算机做迭代运算。

2. 译码准则

信息论方法需要选择信道，然后根据输出判定输入是什么类别。

这里只研究二元信道译码准则，多元信道可以转换为二元信道。二元信道如图 7.3 所示。

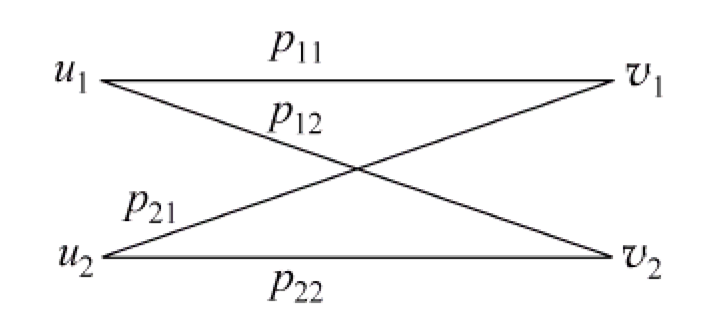


图 7.3 二元信道

将其中转移概率用矩阵表示为 $\begin{bmatrix}p_{11} & p_{12}\\ p_{21} & p_{22}\end{bmatrix}$

举一个简单例子，设有二元信道，其转移概率矩阵为$\boldsymbol{P}=\begin{bmatrix}1/3 & 2/3\\ 2/3 & 1/3\end{bmatrix}$。

当得到特征值 v_1 时，若判定实体的类别为 u_1，则译对的可能性 p_{11} 为 1/3，译错的可能性

p_{21}为 2/3。反之得到 v_1时译成 u_2,则译对的可能性 p_{21}为 2/3,译错的可能性 p_{11}为 1/3。可见译错的概率既与信道的统计特性有关又与译码准则有关。

现在要定义一个译码准则。设信道如图 7.3 所示,定义译码准则就是要设计一个函数 $F(v_j)$对于输出的每一个 v_j惟一确定输入的一个类别 u_i与之对应(单值函数)。

二元信道可以定义译码准则:

$$A:\begin{bmatrix} F(v_1)=u_1 \\ F(v_2)=u_2 \end{bmatrix} \quad \text{或者} \quad B:\begin{bmatrix} F(v_1)=u_2 \\ F(v_2)=u_1 \end{bmatrix}$$

还可以有另外的定义方法。问题是如何得到使平均错误概率最小的译码规则。

(1) 最大后验概率准则

后验概率 $P(u_i/v_j)$表示输入 u_i 发生以后 v_j 出现的概率。用 $P(u_*/v_j)$表示 $P(u_1/v_j)$与 $P(u_2/v_j)$中的一个。

当满足条件 $P(u_*|v_j)\geqslant P(u_i|v_j)$,$i=1,2$ 时定义译码函数 $F(v_j)=u_*$。

其中 u_*是 u_1和 u_2中的某一个。可以证明该准则的平均错误概率最小。即把每个 v_j判成具有最大后验概率 $P(u_i|v_j)$的那个类别。这个准则称为“最大后验概率准则”或“最小错误概率准则”。

(2) 最大似然译码准则

转移概率 $P(v_j|u_*)$是取 $P(v_j/u_1)$与 $P(v_j/u_2)$其中之一。

当满足 $P(v_j|u_*)\geqslant P(v_j|u_i)$,译码函数仍定义为 $F(v_j)=u_*$。这样定义的译码准则称为最大似然译码准则。在 $P(u_1)=P(u_2)$时,两种准则是等价的。

7.2 决策树方法

7.2.1 决策树概念

决策树是用样本的属性作为结点,用属性的取值作为分支的树结构,是利用信息论原理对大量样本的属性进行分析和归纳而产生的。决策树的根结点是所有样本中信息量最大的属性。树的中间结点是该结点为根的子树所包含的样本子集中信息量最大的属性。决策树的叶结点是样本的类别值。

决策树用于对新样本的分类,即通过决策树对新样本属性值的测试,从树的根结点开始,按照样本属性的取值,逐渐沿着决策树向下,直到树的叶结点,该叶结点表示的类别就是新样本的类别。决策树方法是数据挖掘中非常有效的分类方法。

决策树是一种知识表示形式,是对所有样本数据的高度概括,即决策树能准确地识别所有样本的类别,也能有效地识别新样本的类别。

决策树概念最早出现在 CLS(concept learning system)中,影响最大的是 J. R. Quinlan 于 1986 年提出的 ID3 方法,他提出用信息增益(即信息论中的互信息)来选择属性作为决策树的结点。由于决策树的建树算法思想简单,识别样本效率高的特点,使 ID3 方法成为当时机器学习领域中最有影响的方法之一。后来,不少学者提出了改进 ID3 的方法,比较有影响

的是 ID4、ID5 方法。J. R. Quinlan 本人于 1993 年提出了改进 ID3 的 C4.5 方法,C4.5 方法是用信息增益率来选择属性作为决策树的结点,这样建立的决策树识别样本的效率更提高了。C4.5 方法还增加剪枝、连续属性的离散化、产生规则等功能。它使决策树方法再一次得到了提高。

从 ID3 方法到 C4.5 方法,决策树的结点均由单个属性构成,缺少不同属性的关系。我们在研究信息论以后,于 1991 年提出了基于信道容量的 IBLE 方法和 1994 年提出的基于归一化互信息的 IBLE-R 方法。此两方法建立的是决策规则树。树的结点是由多个属性组成。这样,在树的结点中体现了多个属性的相互关系。由于信道容量是互信息的最大值,不随样本数的改变而改变,从而使 IBLE 方法在样本识别效率上,比 ID3 方法提高了 10%。IBLE-R 方法在 IBLE 方法的基础上增加了产生规则的功能。

决策树方法 ID3 和 C4.5 以及决策规则树方法 IBLE 和 IBLE-R 的理论基础都是信息论。

7.2.2 ID3 方法基本思想

J. R. Quinlan 的 ID3 方法的前身是 CLS 方法。Hunt 提出的 CLS 的工作过程为:首先找出有判别力的属性,把数据分成多个子集,每个子集又选择有判别力的属性进行划分,一直进行到所有子集仅包含同一类型的数据为止。最后得到一棵决策树,可以用它来对新的样例进行分类。CLS 的不足是没有说明如何选择有判断力的属性。

J. R. Quinlan 的工作主要是引进了信息论中的互信息,他将其称为信息增益(information gain),作为特征(属性)判别能力的度量,并且将建树的方法嵌在一个迭代的外壳之中。

在一实体世界中,每个实体用多个特征来描述。每个特征限于在一个离散集中取互斥的值。例如,设实体是某天早晨,分类任务是关于气候的类型,特征(属性)为:

- 天气　取值为:晴,多云,雨
- 气温　取值为:冷,适中,热
- 湿度　取值为:高,正常
- 风　　取值为:有风, 无风

每个实体属于不同的类别,为简单起见,假定仅有两个类别,分别为 P、N。在这种两个类别的归纳任务中,P 类和 N 类的实体分别称为概念的正例和反例。将一些已知的正例和反例放在一起便得到训练集。

表 7.1 给出一个训练集。由归纳学习算法 ID3 算法得出一棵正确分类训练集中每个实体的决策树,如图 7.4 所示。该决策树能对训练集中的每个实体按特征取值,判别出它属于 P、N 中的哪一类。

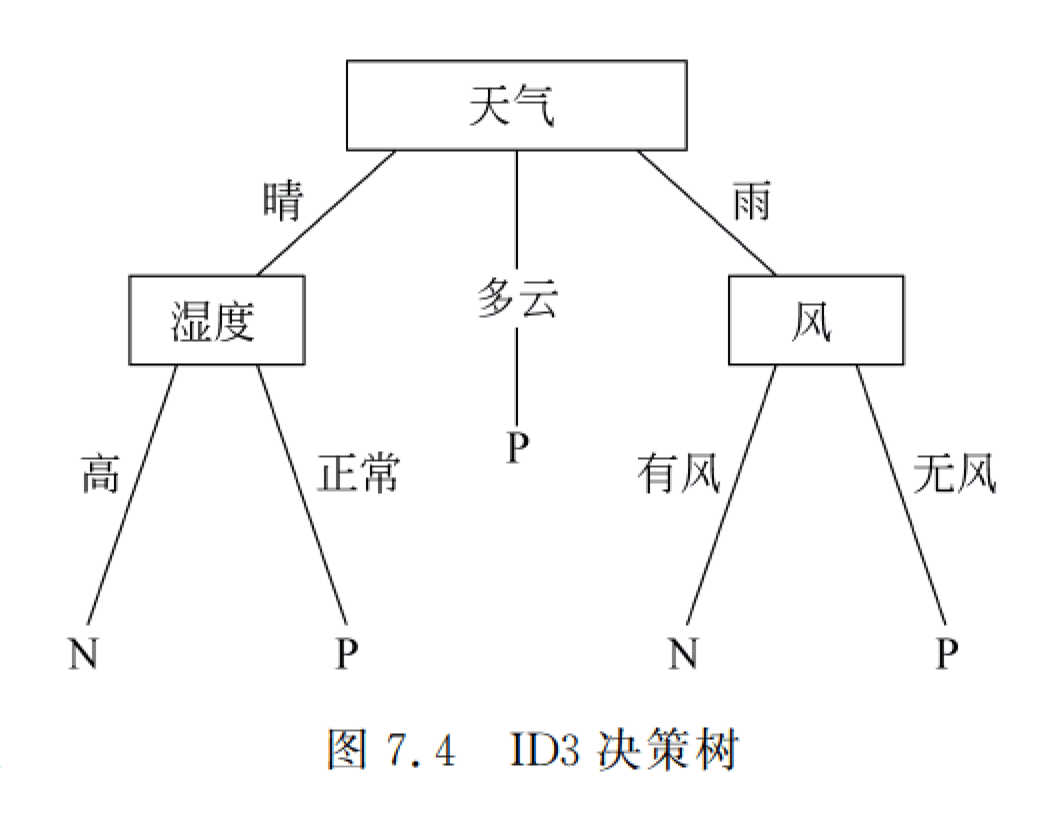


图 7.4　ID3 决策树

表 7.1　气候训练集

No.	属性				类别
	天气	气温	湿度	风	
1	晴	热	高	无风	N
2	晴	热	高	有风	N
3	多云	热	高	无风	P
4	雨	适中	高	无风	P
5	雨	冷	正常	无风	P
6	雨	冷	正常	有风	N
7	多云	冷	正常	有风	P
8	晴	适中	高	无风	N
9	晴	冷	正常	无风	P
10	雨	适中	正常	无风	P
11	晴	适中	正常	有风	P
12	多云	适中	高	有风	P
13	多云	热	正常	无风	P
14	雨	适中	高	有风	N

决策树叶子为类别名，即 P 或者 N。其他结点由实体的特征组成，每个特征的不同取值对应一分支。若要对一实体分类，从树根开始进行测试，按特征的取值分支向下进入下层结点，对该结点进行测试，过程一直进行到叶结点，实体被判属于该叶结点所标记的类别。现有训练集外的一个例子，某天早晨气候描述为：

- 天气：多云
- 气温：冷
- 湿度：正常
- 风：无风

它属于哪类气候呢？用图 7.4 来判别，可以得该实体的类别为 P 类。

实际上，能正确分类训练集的决策树不止一棵。J. R. Quinlan 的 ID3 算法能得出结点最少的决策树。

7.2.3　ID3 算法

1. 主算法

(1) 从训练集中随机选择一个既含正例又含反例的子集(称为“窗口”)；

(2) 用“建树算法”对当前窗口形成一棵决策树；

(3) 对训练集(窗口除外)中例子用所得决策树进行类别判定，找出错判的例子；

(4) 若存在错判的例子，把它们插入窗口，转(2)，否则结束。

主算法流程用图 7.5 表示。其中 PE、NE 分别表示正例集和反例集，它们共同组成训练集。PE′、PE″和 NE′、NE″分别表示正例集和反例集的子集。

主算法中每迭代循环一次，生成的决策树将会不相同。

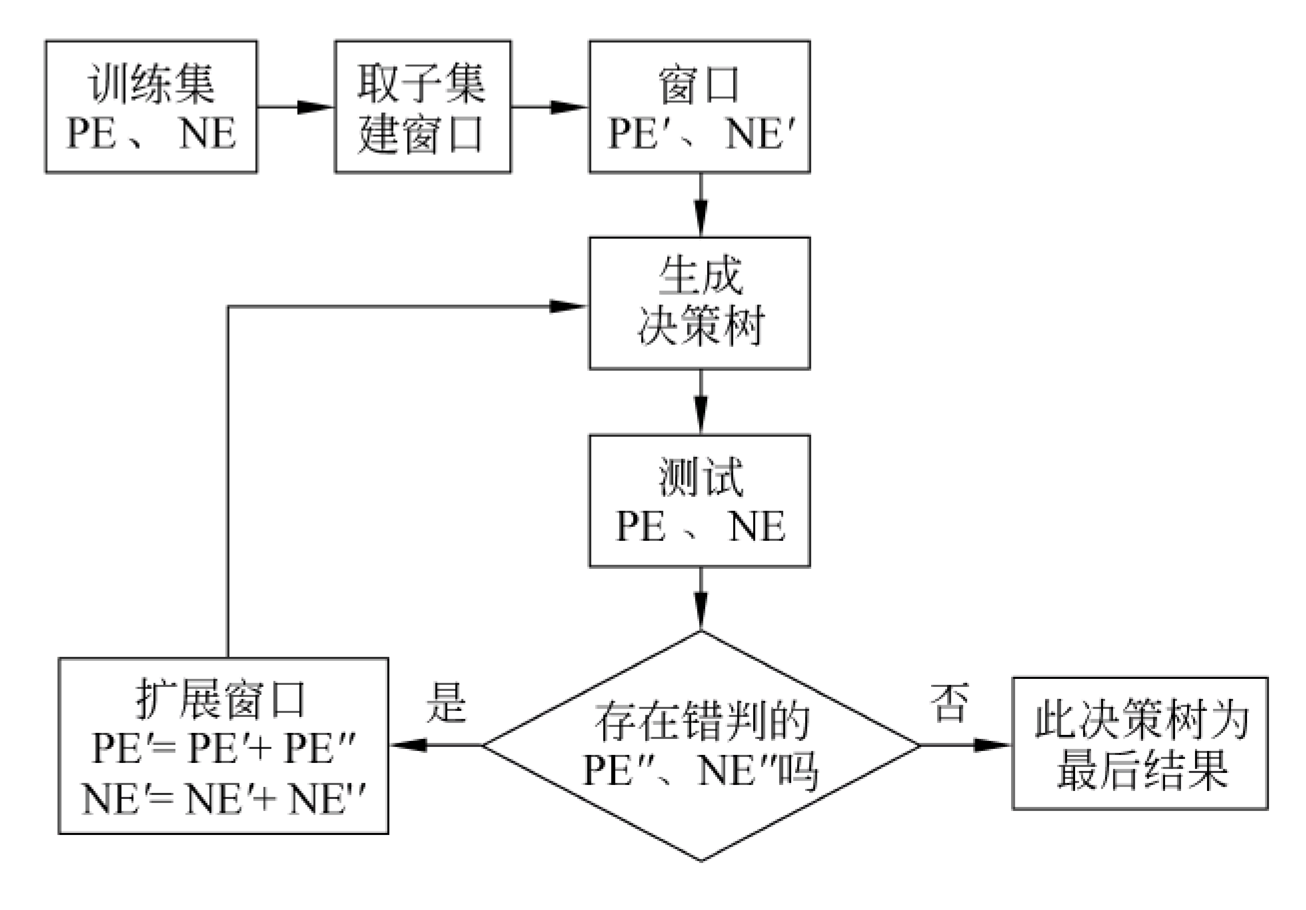


图 7.5　ID3 主算法流程

2. 建树算法

(1) 对当前例子集合计算各特征的互信息。

(2) 选择互信息最大的特征 A_k 作为树(或子树)的根结点。

(3) 把在 A_k 处取值相同的例子归于同一子集,该取值作为树的分支。A_k 取几个值就得几个子集,各取值作为树的一个分支。

(4) 对既含正例又含反例的子集递归调用建树算法。

(5) 若子集仅含正例或反例,对应分支标上 P 或 N,返回调用处。

7.2.4　实例与讨论

1. 实例计算

对于气候分类问题进行具体计算如下。

(1) 信息熵的计算

信息熵:$H(U) = -\sum_i P(u_i)\log_2 P(u_i)$

类别 u_i 出现概率:$P(u_i) = \dfrac{|u_i|}{|S|}$

$|S|$表示例子集 S 的总数,$|u_i|$表示类别 u_i 的例子数。

对 9 个正例和 5 个反例有:$P(u_1) = 9/14$,$P(u_2) = 5/14$

$$H(U) = (9/14)\log_2(14/9) + (5/14)\log_2(14/5) \approx 0.94 \text{ b}$$

(2) 条件熵计算

条件熵:$H(U \mid V) = -\sum_j P(v_j) \sum_i P(u_i/v_j)\log_2 P(u_i/v_j)$

属性 A_1 取值 v_j 时,类别 u_i 的条件概率:$P(u_i/v_j) = \dfrac{|u_i|}{|v_j|}$

A_1 = 天气,它的取值有:v_1 = 晴,v_2 = 多云,v_3 = 雨。

在 A_1 处取值“晴”的例子有 5 个,取值“多云”的例子有 4 个,取值“雨”的例子有 5 个,故:

$$P(v_1) = 5/14,\quad P(v_2) = 4/14,\quad P(v_3) = 5/14$$

取值为“晴”的5个例子中有2个正例、3个反例，故：

$$P(u_1/v_1) = 2/5,\quad P(u_2/v_1) = 3/5$$

取值为“多云”时有：$P(u_1/v_2)=4/4$，　$P(u_2/v_2)=0$

取值为“雨”时有：$P(u_1/v_3)=2/5$，$P(u_2/v_3)=3/5$

$$H(U \mid V) = (5/14)((2/5)\log(5/2) + (3/5)\log(5/3)) + (4/14)((4/4)\log(4/4) + 0) + (5/14)((2/5)\log(5/2) + (3/5)\log(5/3)) \approx 0.694 \text{ b}$$

(3) 互信息计算

对 A_1＝天气处有：

$$I(\text{天气}) = H(U) - H(U \mid V) = 0.94 - 0.694 \approx 0.246 \text{ b}$$

类似可得：

$I(\text{气温})=0.029$ b

$I(\text{湿度})=0.151$ b

$I(\text{风})=0.048$ b

(4) 建决策树的树根和分支

ID3算法将选择互信息最大的特征“天气”作为树根，在14个例子中对“天气”的3个取值进行分支，3个分支对应3个子集，分别是：

$$F1 = \{1,2,8,9,11\},\quad F2 = \{3,7,12,13\},\quad F3 = \{4,5,6,10,14\}$$

其中 $F2$ 中的例子全属于P类，因此对应分支标记为P，其余两个子集既含有正例又含有反例，将递归调用建树算法。

(5) 递归建树

分别对 $F1$ 和 $F3$ 子集利用ID3算法，在每个子集中对各特征(仍为4个特征)求互信息。

① $F1$ 中的“天气”全取“晴”值，则 $H(U)=H(U|V)$，有 $I(U|V)=0$，在余下3个特征中求出“湿度”互信息最大，以它为该分支的根结点，再向下分支。“湿度”取“高”的例子全为N类，该分支标记N。取值“正常”的例子全为P类，该分支标记P。

② 在 $F3$ 中，对4个特征求互信息，得到“风”特征互信息最大，则以它为该分支根结点。再向下分支，“风”取“有风”时全为N类，该分支标记N。取“无风”时全为P类，该分支标记P。

这样就得到图7.4的决策树。

2. 对ID3的讨论

(1) 优点

ID3在选择重要特征时利用了互信息的概念，算法的基础理论清晰，使得算法较简单，是一个很有实用价值的示例学习算法。

该算法的计算时间是例子个数、特征个数、结点个数之积的线性函数。我们曾用4761个关于苯的质谱例子做了试验。其中正例2361个，反例2400个，每个例子由500个特征描述，每个特征取值数目为6，得到一棵1514个结点的决策树。对正、反例各100个测试例做了测试，正例判对82个，反例判对80个，总预测正确率为81%，效果是令人满意的。

(2) 缺点

① 互信息的计算依赖于特征取值的数目较多的特征，这样不太合理。一种简单的办法是对特征进行分解，如上节例中，特征取值数目不一样，可以把它们统统化为二值特征，如天气取值晴、多云、雨，可以分解为 3 个特征：天气-晴，天气-多云，天气-雨。取值都为“是”或“否”，对气温也可做类似的工作。这样就不存在偏向问题了。

② 用互信息作为特征选择量存在一个假设，即训练例子集中的正、反例的比例应与实际问题领域里正、反例比例相同。一般情况下不能保证相同，这样计算训练集的互信息就有偏差。

③ ID3 在建树时，每个结点仅含一个特征，是一种单变元的算法，特征间的相关性强调不够。虽然它将多个特征用一棵树连在一起，但联系还是松散的。

④ ID3 对噪声较为敏感。关于什么是噪声，J. R. Quinlan 的定义是训练例子中的错误就是噪声。它包含两方面，一是特征值取错，二是类别取错。

⑤ 当训练集增加时，ID3 的决策树会随之变化。在建树过程中，各特征的互信息会随例子的增加而改变，从而使决策树也变化。这对渐近学习(即训练例子不断增加)是不方便的。

总的来说，ID3 由于其理论的清晰，方法简单，学习能力较强，适于处理大规模的学习问题，并广为流传，受到极大的关注，是数据挖掘和机器学习领域中的一个极好范例，也不失为一种知识获取的有用工具。

7.2.5 C4.5 方法

ID3 算法在数据挖掘中占有非常重要的地位。但是，在应用中，ID3 算法不能够处理连续属性、计算信息增益时偏向于选择取值较多的属性等不足。C4.5 是在 ID3 基础上发展起来的决策树生成算法，由 J. R. Quinlan 在 1993 年提出。C4.5 克服了 ID3 在应用中存在的不足，主要体现在以下几个方面：

(1) 用信息增益率来选择属性，克服了用信息增益选择属性时偏向选择取值多的属性的不足；

(2) 在树构造过程中或者构造完成之后，进行剪枝；

(3) 能够完成对连续属性的离散化处理；

(4) 能够对于不完整数据的处理，例如未知的属性值；

(5) C4.5 采用的知识表示形式为决策树，并最终可以形成产生式规则。

1. 构造决策树

设 T 为数据集，类别集合为 $\{C_1, C_2, \cdots, C_k\}$，选择一个属性 V 把 T 分为多个子集。设 V 有互不重合的 n 个取值 $\{v_1, v_2, \cdots, v_n\}$，则 T 被分为 n 个子集 $T_1, T_2, \cdots, T_n$，这里 T_i 中的所有实例的取值均为 v_i。

令：$|T|$ 为数据集 T 的例子数，$|T_i|$ 为 $v=v_i$ 的例子数，$|C_j| = \text{freq}(C_j, T)$ 为 C_j 类的例子数，$|C_j^v|$ 是 $V=v_i$ 例子中具有 C_j 类别例子数。

则有：

① 类别 C_j 的发生概率：$p(C_j)=|C_j|/|T| = \text{freq}(C_j,T)/|T|$

② 属性 $V=v_i$ 的发生概率：$p(v_i) = |T_i|/|T|$

③ 属性 $V=v_i$ 的例子中具有类别 C_j 的条件概率：

$$p(C_j \mid v_i) = |C_j^v| / |T_i|$$

Quinlan 在 ID3 中使用信息论中的信息增益(gain)来选择属性，而 C4.5 采用属性的信息增益率(gain_ratio)来选择属性。

以下公式中的 $H(C)$、$H(C/V)$、$I(C,V)$、$H(V)$ 是信息论中的写法，而 info(T)、info$_v(T)$、gain(V)、plit_info(V)、gain_ratio 是 Quinlan 的写法，在此统一起来。

(1) 类别的信息熵

$$H(C) = -\sum_{j=1}^{k} p(C_j)\log(p(C_j)) = -\sum_{j=1}^{k}\frac{|C_j|}{|T|}\log\left(\frac{|C_j|}{|T|}\right)$$

$$= -\sum_{j=1}^{k}\frac{\text{freq}(C_j,T)}{|T|}\times\log_2\left(\frac{\text{freq}(C_j,T)}{|T|}\right) = \text{info}(T)$$

(2) 类别条件熵

按照属性 V 把集合 T 分割，分割后的类别条件熵为

$$H(C \mid V) = -\sum_{j} p(v_j)\sum_{i} p(C_j \mid v_i)\log p(C_j \mid v_i)$$

$$= -\sum_{j}\frac{|T_j|}{|T|}\sum_{i}\frac{|C_j^v|}{|T_i|}\log\frac{|C_j^v|}{|T_i|}$$

$$= \sum_{i=1}^{n}\frac{|T_i|}{|T|}\text{info}(T_i) = \text{info}_v(T)$$

(3) 信息增益(gain)，即互信息

$$I(C,V) = H(C) - H(C \mid V) = \text{info}(T) - \text{info}_v(T) = \text{gain}(V)$$

(4) 属性 V 的信息熵

$$H(V) = -\sum_{i} p(v_i)\log(p(v_i)) = -\sum_{i=1}^{n}\frac{|T_i|}{|T|}\log_2\left(\frac{|T_i|}{|T|}\right) = \text{split_info}(V)$$

(5) 信息增益率

$$\text{gain_ratio} = I(C,V)/H(V) = \text{gain}(V)/\text{split_info}(V)$$

C4.5 方法对 ID3 的改进是用信息增益率来选择属性。

理论和实验表明，采用“信息增益率”(C4.5 方法)比采用“信息增益”(ID3 方法)更好，主要是克服了 ID3 方法选择偏向取值多的属性。

2. 连续属性的处理

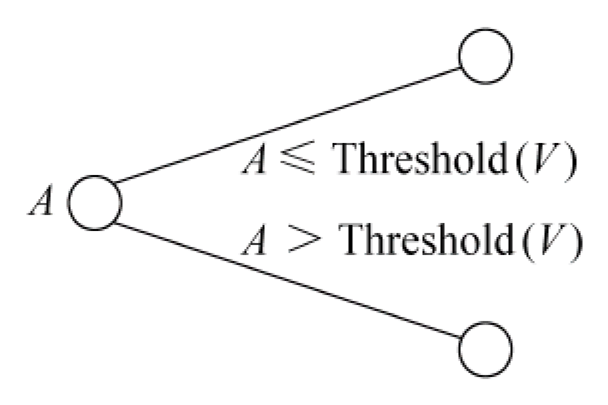
在 ID3 中没有处理连续属性的功能。在 C4.5 方法中，设在集合 T 中，连续属性 A 的取值为 $\{v_1,v_2,\cdots,v_m\}$，则任何在 v_i 和 v_{i+1} 之间的任意取值都可以把实例集合分为两部分：$T_1=\{t|A\leqslant v_i\}$ 和 $T_2=\{t|A>v_i\}$。

可以看到一共有 $m-1$ 种分割情况，对属性 A 的 $m-1$ 种分割的任意一种情况，作为该属性的两个离散取值，重新构造该属性的离散值，再按照上述公式计算每种分割所对应的信息增益率 gain_ratio(v_i)，在 $m-1$ 种分割中，选择最大增益率的分割作为属性 A 的分支，即

$$\text{Threshold}(V) = v_k$$

其中，$\text{gain_ratio}(v_k) = \max\{\text{gain_ratio}(v_i)\}$，即 v_k 是各 v_i 的信息增益率最大者。

则连续属性 A 可以分割为：



3. 决策树剪枝

由于噪声和随机因素的影响，决策树一般会很复杂，因此需要进行剪枝操作。

（1）什么时候剪枝

有两种剪枝策略：

① 在树生成过程中判断是否还继续扩展决策树。若停止扩展，则相当于剪去该结点以下的分支。

② 对于生成好的树剪去某些结点和分支。C4.5 方法采用第二种方法。

剪枝之后的决策树的叶结点不再只包含一类实例。结点有一个类分布描述，即该叶结点属于某类的概率。

（2）基于误差的剪枝

决策树的剪枝通常是用叶结点替代一个或者多个子树，然后选择出现概率最高的类作为该结点的类别。在 C4.5 方法中，还允许用其中的树枝来替代子树。

如果使用叶结点或者树枝代替原来的子树之后，误差率若能够下降，则使用此叶结点或者树枝代替原来的子树。

4. 从决策树抽取规则

在 C4.5 方法中，对于生成的决策树，可以直接获得规则。从根到叶的每一条路径都可以是一条规则。这样，可以看出有多少条路径就可以产生多少条规则。例如，从下面的决策树中可以得到规则：

决策树：

```
| F=0
|    J=0: Class0
|    J=1
|    |  K=0: Class0
|    |  K=1: Class1
| F=1
|    G=1: Class1
|    G=0
|    |  J=0: Class0
|    |  J=1
|    |  |  K=0: Class0
|    |  |  K=1: Class1
```

沿着决策树的其中一条路径 F → G → J → K 得到规则：

IF F=1，G=0，J=1，K=1 THEN Class1

7.3　决策规则树方法

7.3.1　IBLE 方法的基本思想

1. IBLE 方法的特点

我们于 1991 年研制的 IBLE 方法是利用信息论中信道容量的概念作为对实体中选择重要特征的度量。信道容量是一个不依赖于正、反例的比例，仅依赖于训练集中正、反例的特征取值的选择量。这样，信道容量克服了互信息依赖正、反例比例的缺点。IBLE 方法不同于 ID3 方法每次只选一个特征作为决策树的结点，而是选一组重要特征建立规则作为决策树的结点。这样，用多个特征组合成规则的结点来鉴别实例，能够更有效地正确判别。对那些不能直接判定的例子继续利用决策规则树的其他规则结点来判别，这样一直进行下去，直至判出类别为止。

IBLE 方法建立的是决策规则树，树中每个结点是由多个特征所组成。特征的选取是通过计算各特征信道容量来进行的。各特征的正例标准值由译码函数决定。结点中判别正、反例的阈值(S_p，S_n)是由实例中权值变化的规律来确定的。

2. 多元信道转化成二元信道

在各特征取多值的情况下，用互信息作为特征选择量，会出现倾向于取某值的例子数较多的特征，这种倾向并不都合理。用信道容量作为特征选择量也必然有同样的问题存在。一种解决办法是对特征进行分解，如前面举的例中，如果特征取值数目不一样，则可以把它们统统化为二值特征。如天气取值晴、多云、雨，可以分解成 3 个特征：天气-晴、天气-多云、天气-雨，每个都取值为{yes，no}，对气温也可以做类似的工作。这样在选择特征时就不会出现偏向问题了。

3. 决策规则树

IBLE 是基于信息论的示例学习方法(information-based learning from examples，IBLE)。IBLE 算法从训练集归纳出一棵决策规则树。

判定一个实体属于 u_1类，还是属于 u_2类，首先从分析该实体的特征入手，用规则分析会得出 3 种可能结论：①该实体属于 u_1类，②该实体属于 u_2类，③不能作出判定，需进一步分析再做结论。在进一步分析时又会出现上述 3 种情形。对一实体的分析，这个过程一直进行到得出具体类别为止。IBLE 就是依据这种思想构造决策规则树的。决策规则树如图 7.6 所示。

对于更复杂的问题除使用主规则外，还增加分规则，得出如图 7.7 所示的决策规则树。

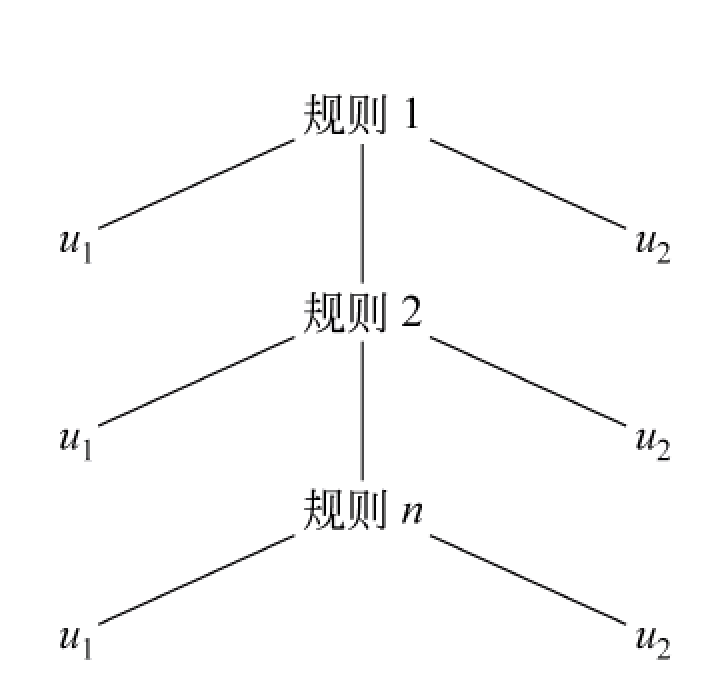


图 7.6　IBLE 算法的一般决策规则树

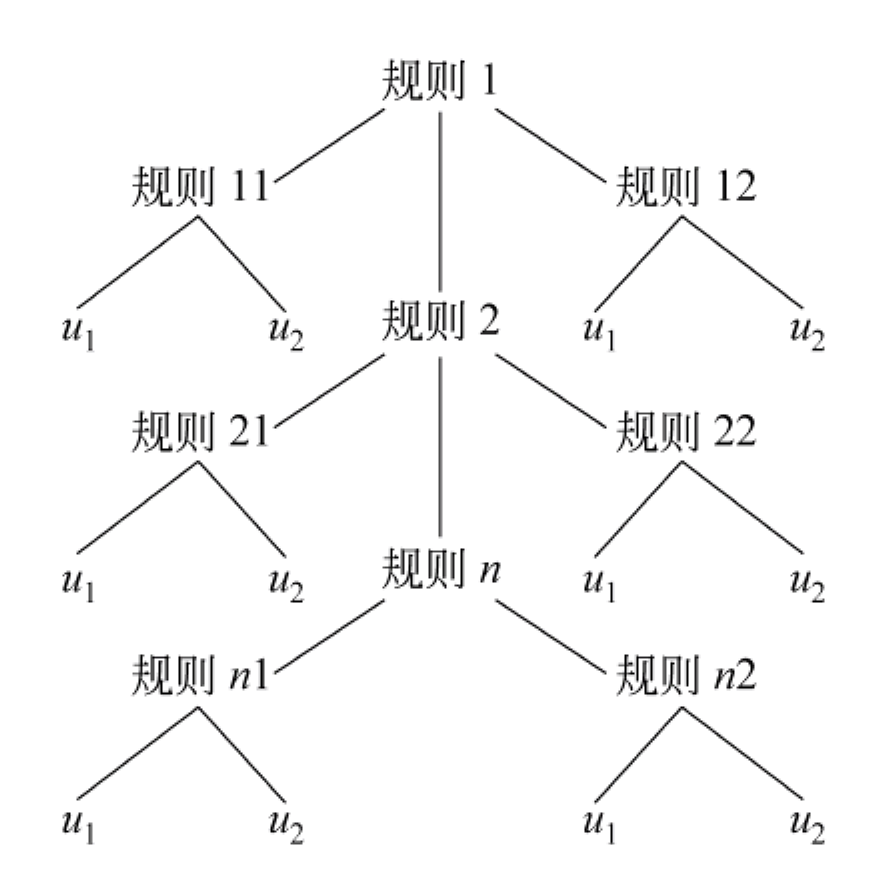


图 7.7　IBLE 算法的复杂决策规则树

4. 决策规则树结点

(1) 规则表示形式

决策规则树中非叶结点均为规则。规则表示为：

- 特征：$A_1, A_2, \cdots, A_m$
- 权值：$W_1, W_2, \cdots, W_m$
- 标准值：$V_1, V_2, \cdots, V_m$
- 阈值：S_p, S_n

该规则可形式描述为：

① sum：=0；

② 对 i：=1 到 m 作：若$(A_i)=V_i$，则 sum：=sum$+W_i$；

③ 若 sum$\leqslant S_n$，则该例为 N 类；

④ 若 sum$\geqslant S_p$，则该例为 P 类；

⑤ 若 $S_n<$sum$<S_p$，则该例暂不能判，转下一条规则判别。

其中 sum 表示权和，(A_i)表示特征 A_i的取值。

(2) 举例

下面为说明规则中各成分的意义，举一个例子。设问题空间中例子有 10 个特征(属性)，特征编号从 1 到 10。每个特性取值为{no，yes}，用{0，1}表示，规则是由重要特征组成的，对每个特征求出权值以表示其重要程度，删除不重要特征，得规则如下：

- 特征：　1　3　4　6　7
- 权值：　100　90　105　500　40
- 标准值：　1　0　1　1　0
- 阈值：220，100

现有 3 个测试例子：

例子 1：(1，0，0，0，1，0，0，1，1，1)

例子 2：(0，1，0，0，1，0，0，0，1，0)

例子 3：(0,1,0,0,1,0,1,0,1,1)

例子 1 的权和 sum=230，有 sum>220，判定例子 1 属于 u_1类。例子 2 的权和 sum=130，有 100<sum<220，认为例子 2 不能判，而例子 3 有权和 sum=90，有 sum<100，判例子 3 的类别为 u_2类。

通过上例知道规则中 $A_1,A_2,\cdots,A_m$为组成规则的特征，$W_1,W_2,\cdots,W_m$为对应的权值，$V_1,V_2,\cdots,V_m$为对应特征取正例的标准值，若例子在该特征处取值与标准值相同，则 sum（权和）加上对应权值，否则不加。S_p、S_n是判是、判非、不能判的阈值。若例子的权和为 sum，sum$\geqslant S_p$时判为是类（u_1类），sum$\leqslant S_n$时判为非类（u_2类），$S_n<$sum$<S_p$时认为不能判。由于 S_p、S_n的作用已知，分规则中必有 $S_p=S_n$。

7.3.2 IBLE 算法

IBLE 算法由 4 部分组成：预处理；建决策树算法；建规则算法；类别判定算法。下面分别介绍。

1. 预处理

将例子集的特征取多值，变为多个特征分别取{0,1}值。即一个特征取 n 个值变为 n 个特征分别取{0,1}值。

2. 建规则算法

- 求各特征 A_k的信道容量 C_k，对于一个特征有分特征（原一个特征取多值变成多个特征取{0,1}值时，该多个特征为原特征的分特征）时，取最大 C 值的分特征代表该特征。

权值的计算（取整）公式为 $W_k=[C_k\times1000]$。

- 利用最大后验准则定义该特征 A_k 的译码函数 $F(1)$、$F(0)$。

设类别为 u_1、u_2，特征 V 取值 1 和 0，转移概率为 $P(1/u_1)$、$P(0/u_1)$、$P(1/u_2)$、$P(0/u_1)$。信道容量计算后，可同时得到类别的先验概率 $P(u_1)$和 $P(u_2)$。于是，令 sum$=P(u_1)\times P(1/u_1)+P(u_2)\times P(1/u_2)$。由贝叶斯公式：$P(u_1/1)=P(u_1)\times P(1/u_1)/$sum，$P(u_2/1)=P(u_2)\times P(1/u_2)/$sum。译码准则为：当 $P(u_1/1)\geqslant P(u_2/1)$时，$F(1)=u_1$；否则，$F(0)=u_1$。这样，就定义了特征 V 对类别 u_1（正例）的标准值 1 或 0。可以证明，该准则的错误概率最小。

- 利用译码函数按正例（u_1）输入，计算特征 A_k的标准值{0,1}。
- 选取前 m 个信道容量（即权值）较大的特征构造规则。

一般来说，m 的选取应保证 $C>0.01$b 的特征都被选中（对具体问题可通过试验来确定）。

- 计算所有的正、反例的权和数，从它们的分布规律中得出 S_p、S_n阈值。

建立一个二维数组 $A(m,n)$，$m=1,2,3$；$n=1,2,\cdots,|U|$（$|U|$表示例子总数）。它由三项组成：$A(1,n)$存放各例的权和（例子中各特征的权值累加之和）；$A(2,n)$存放正例个数，当例子是正例时，它为 1，反之为零；$A(3,n)$存放反例个数，当例子是反例时，它为 1，反之

为零。

先对各正、反例子求权和并填入数组 $A(m,n)$ 中。再按权和大小从小到大的顺序对数组 $A(m,n)$ 进行排序，对权和相同的不同的正、反例，将它们合并成一列相同的权和，累计正、反例个数。这样，数组缩小了，即 $n \leqslant |U|$。而且正、反例权和的规律性就出现了：权和小的部分，正例个数为零，反例个数偏大；权和大的部分，正例个数偏大，反例个数为零，如图 7.8 所示。

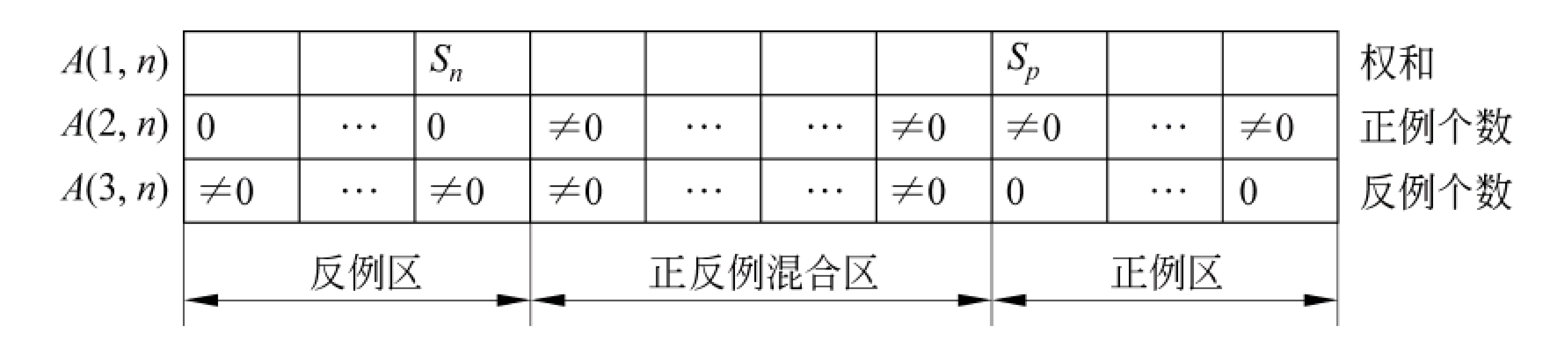


图 7.8　正、反例权和变化规律

从图 7.8 中可知，整个例子集合中划分成 3 个区：反例区、正反例混合区、正例区。在反例区中，正例个数 $A(2,n)$ 均为零。在正例区中，反例个数 $A(3,n)$ 均为零。在混合区中，正例个数 $A(2,n)$ 和反例个数 $A(3,n)$ 均不为零。在 3 个区的分界线处的权和值作为 S_p、S_n 值，用作判别正反例的阈值。

3. 建决策树算法

设 T 为存放决策规则树的空间。建决策树算法如下：

(a) 置决策规则树 T 为空。分配一新结点 R，$T:=R$；

(b) 对当前训练集 PE∪NE，利用“建规则算法”构造主规则；

(c) 用当前规则测试 PE、NE 得子集 PEP、PEN、PEM（正例 3 个子集），以及 NEP、NEN、NEM（反例 3 个子集）。其中 PEP、PEN、PEM 分别表示正例被判为 P 类、N 类、不能判这 3 个子集，NEP、NEN、NEM 分别表示反例被判为 P 类、N 类、不能判这 3 个子集；

(d) 将当前规则放入结点 R；

(e) 若 $(|PEP| \neq 0) \vee (|NEP| \neq 0)$ 则 PE：=PEP，NE：=NEP；分配一新结点 W_1；R 左指针指向 W_1；

① 对当前训练集 PE∪NE 利用“建规则算法”构造左分规则；

② 将左分规则放入结点 W_1。

(f) 若 $(|PEN| \neq 0) \vee (|NEN| \neq 0)$ 则 PE：=PEN，NE：=NEN；分配一新结点 W_2；R 右指针指向 W_2；

① 对当前训练集 PE∪NE 利用“建规则算法”构造右分规则；

② 将右分规则放入结点 W_2。

(g) 若 $(|PEM| \neq 0) \vee (|NEM| \neq 0)$ 则 PE：=PEM，NE：=NEM；分配一新结点 W_3；R 的中指针指向 W_3；$R:=W_3$；转(b)；

(h) 结束。

建决策树算法如图 7.9 所示。

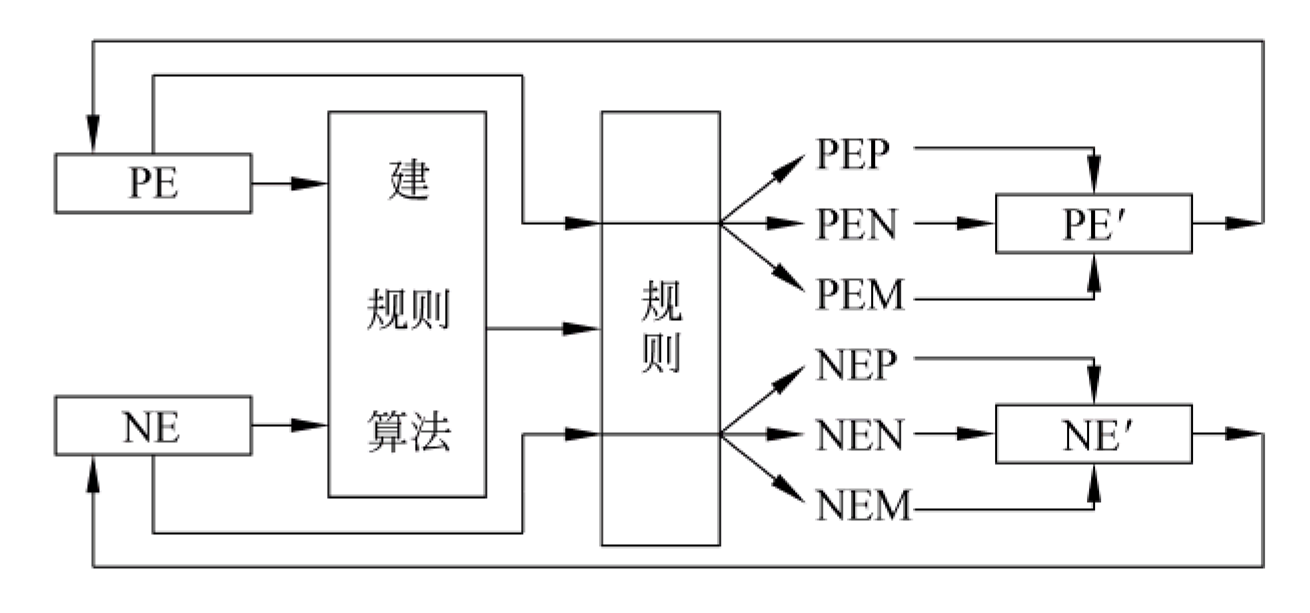


图 7.9 IBLE 建决策树算法图

4. 类别判定算法

在得到一棵决策规则树后,对一未知实体 E 如何分类,下面给出具体的算法:

(1) 置根结点为当前结点;

(2) 用当前结点中的规则对 E 进行判定;

① 判为 P 时(对主规则,该实体不一定是 P 类),若当前结点左指针不空(即左规则存在),将左指针指示的结点置为当前结点且转(2),否则(左指针为空,该实体判为 P 类)转(3);

② 判为 N 时(对主规则,该实体不一定是 N 类),若当前结点右指针不为空(即右规则存在),则将右指针指示的结点置为当前结点且转(2),否则(右指针为空,该实体判为 N 类)转(3);

③ 不能判时,将当前结点的中指针指示的结点置为当前结点转(2)。

(3) 输出判别结果,结束。

7.3.3 IBLE 方法实例

7.3.3.1 配隐形眼镜问题

1. 简例说明

(1) 患者配隐形眼镜的类别

患者是否应配隐形眼镜有 3 类:

@1:患者应配隐形眼镜;

@2:患者应配软隐形眼镜;

@3:患者不适合配隐形眼镜。

(2) 患者眼镜诊断信息(属性)

a:患者的年纪

年轻;前老光眼;老光眼

b:患者的眼睛诊断结果

近视;远视

c:是否散光

是;否

d：患者的泪腺

不发达;正常

(3) 配隐形眼镜实例

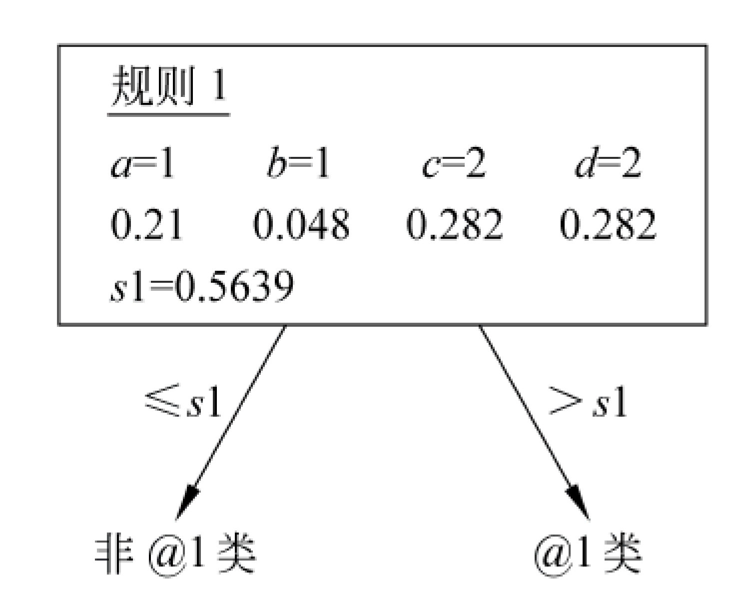
现有 24 个患者实例分别属于 3 个类别,如表 7.2 所示。

表 7.2 配隐形眼镜患者实例

序号	属性取值				诊断值	序号	属性取值				诊断值
	a	b	c	d	@		a	b	c	d	@
1	1	1	1	1	3	13	2	2	1	1	3
2	1	1	1	2	2	14	2	2	1	2	2
3	1	1	2	1	3	15	2	2	2	1	3
4	1	1	2	2	1	16	2	2	2	2	3
5	1	2	1	1	3	17	3	1	1	1	3
6	1	2	1	2	2	18	3	1	1	2	3
7	1	2	2	1	3	19	3	1	2	1	3
8	1	2	2	2	1	20	3	1	2	2	1
9	2	1	1	1	3	21	3	2	1	1	3
10	2	1	1	2	2	22	3	2	1	2	2
11	2	1	2	1	3	23	3	2	2	1	3
12	2	1	2	2	1	24	3	2	2	2	3

2. 利用 IBLE 算法得出的各类决策规则树和逻辑公式

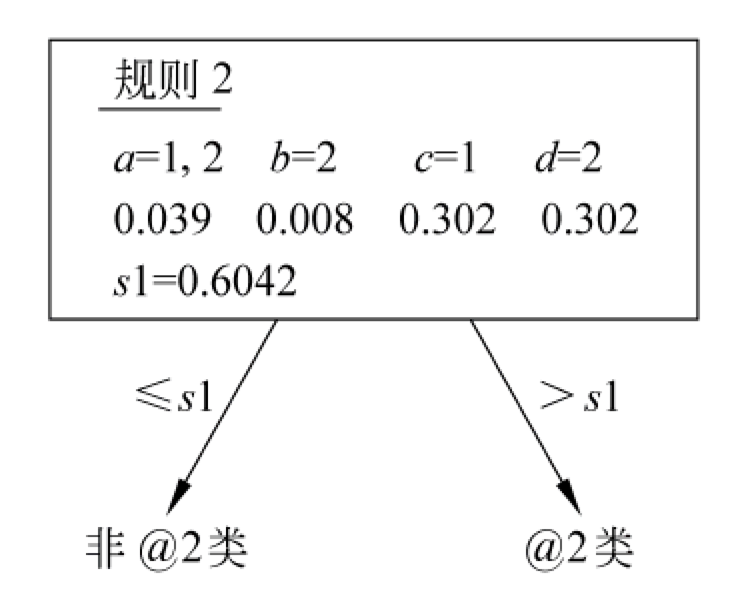
(1) @1 类的决策规则树



相应的逻辑公式为：

$$c = 2 \wedge d = 2 \wedge a = 1 \quad \rightarrow @1$$
$$c = 2 \wedge d = 2 \wedge b = 1 \quad \rightarrow @1$$

(2) @2 类的决策规则树



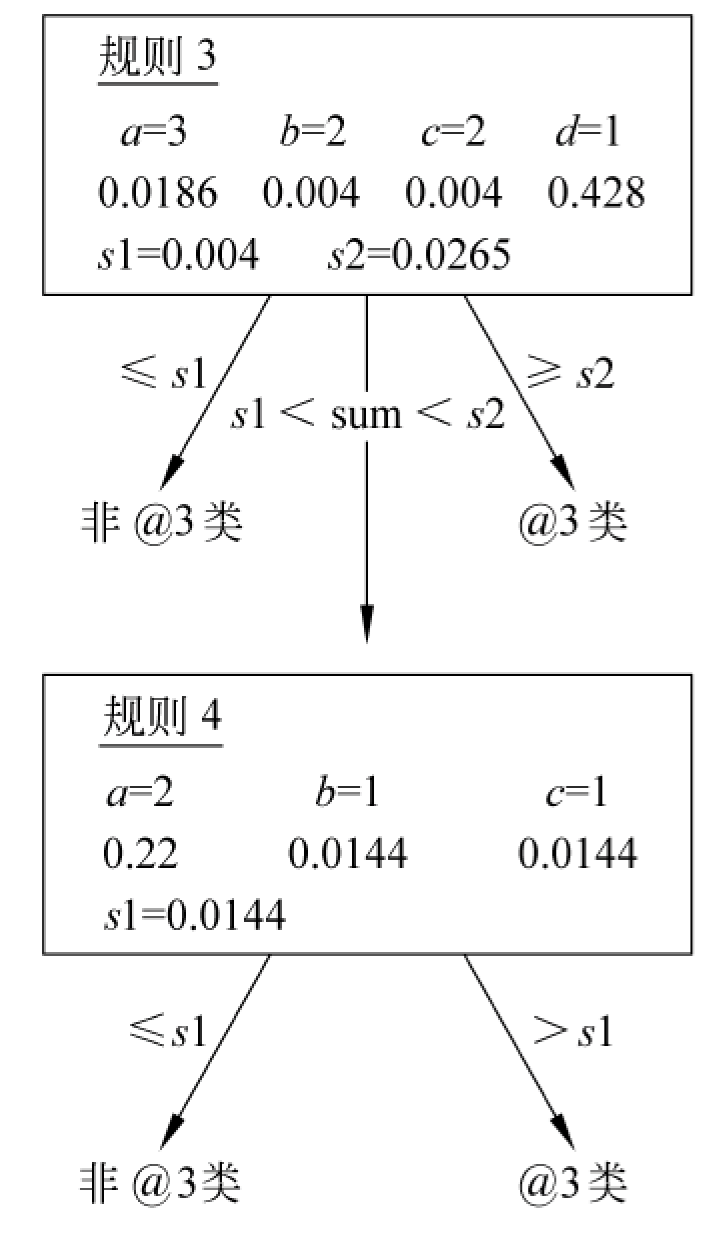
相应的逻辑公式为：

$$c = 1 \wedge d = 2 \wedge b = 2 \quad \rightarrow @2$$

$$c = 1 \wedge d = 2 \wedge a = 1 \quad \rightarrow @2$$

$$c = 1 \wedge d = 2 \wedge a = 2 \quad \rightarrow @2$$

（3）@3 类的决策规则树



该决策树的逻辑公式推导为：

- 上层结点的逻辑公式

$$d = 1 \rightarrow @3$$

$$a = 3 \wedge b = 2 \wedge c = 2 \rightarrow @3$$

- 上层不能判断逻辑公式(中线结论)

$$(b = 2 \wedge c = 2) \vee$$

$$(a = 3) \vee$$

$$(a = 3 \wedge b = 2) \vee$$

$$(a = 3 \wedge c = 2) \rightarrow \text{继续判别}$$

- 下层结点的逻辑公式

$$b = 1 \wedge c = 1 \rightarrow @3$$

$$a = 2 \rightarrow @3$$

- 合并后下层结点的逻辑公式(上层继续判别逻辑公式与下层结点的逻辑公式的合并)

$$a = 3 \wedge b = 1 \wedge c = 1 \rightarrow @3$$

$$a = 2 \wedge b = 2 \wedge c = 2 \rightarrow @3$$

7.3.3.2 苯等 8 类化合物的分类问题

1. **质谱分析**

质谱仪是一种化学分析仪器，以高速电子轰击被测样本，使分子产生分裂碎片且重新排列，测量这些碎片的荷质比即能量形成质谱，如图 7.10 所示。分析化学家根据质谱可以推测出样本的分子结构及性质。这是一个极为复杂和困难的任务，原因在于质谱数据量太大且伴随噪声，而且质谱测定理论尚不完备。在这样的背景下要用传统的知识获取技术建造一个质谱解析专家系统是极为困难的。因此，用计算机从大量的质谱数据中自动获得一些知识便成了一个诱人的设想。

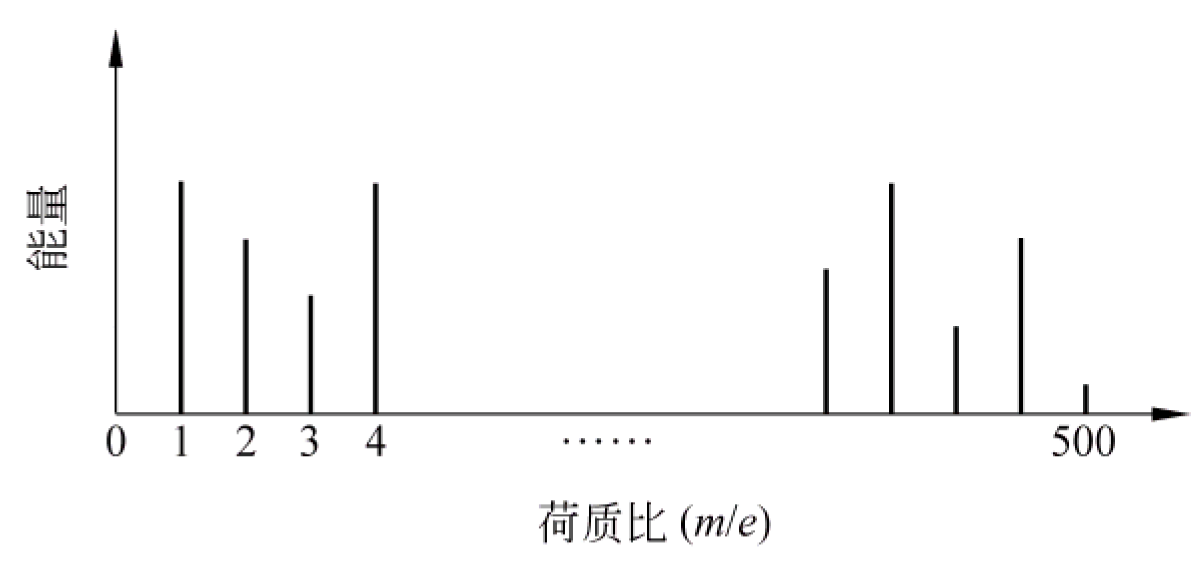


图 7.10 化合物质谱图

2. 实例计算

我们对 8 种类型的化合物进行学习、识别。其中前 3 种类型分别为 WLN 码中含 R、T60TJ 和 QR 的化合物；后 5 种为日内瓦国际会议的技术报告中给出的 5 类有机磷化合物，前 3 种类型化合物的训练集、测试集的构造方法是，从 31231 例质谱中选出某类所有化合物的集合 T_1，剩余的两类成为集合 T_2。从 T_1 中随机抽出一定数目的化合物构成两个集合 T_{11}、T_{12}，再从 T_2 中随机抽取一定数目的化合物构成两个集合 T_{21}、T_{22}，用 T_{11} 和 T_{21} 组成训练集，正例 PE$=T_{11}$，反例 NE$=T_{21}$，用 T_{12} 和 T_{22} 组成测试集。对于后 5 种有机磷化合物，上述 31231 例前 3 类质谱中都没有，按类输入，每种抽取 8 例作为训练集中的正例集，剩下的作为测试集的正例，再从 31231 例质谱中抽出 999 例作为训练集反例集，得出如表 7.3 所示的训练集、测试集。用 IBLE 学习后得出 8 棵决策规则树（在此省略），对测试集进行识别，预测正确率如表 7.4 所示。

本实验中，预测正确率是这样计算的，先分别计算正、反例的预测正确率，然后两者相加除以 2 得出总预测正确率，这种做法在实际问题中可信程度较高。从表 7.5 知道，对 8 类化合物，IBLE 的平均预测正确率为 93.967%。

3. IBLE 与 ID3 的比较

(1) 实例计算情况

为了比较 IBLE 与 ID3 在正、反例数目变化情况下的性能，从 8 种类型中随机抽取 3 类，即 R、T60TJ 和有机磷化合物中的第二类进行实验。两种算法关于 3 种化合物的平均预测正确率如表 7.5 所示。可以看出，预测正确率 IBLE 比 ID3 高出近 10%。

表 7.3　8 类训练物的训练集和测试集

类	训练集		测试集	
	正例	反例	正例	反例
R	2363	2400	102	155
QR	571	2000	20	100
T60TJ	500	2300	50	50
类一	8	999	5	999
类二	8	999	5	999
类三	8	999	2	999
类四	8	999	4	999
类五	8	999	1	999

表 7.4　IBLE 对 8 类化合物的预测结果

类	正例	认对	认错	正确/%	反例	认对	认错	正确/%	总正确/%
R	102	95	7	93.137	155	136	19	87.774	90.439
QR	20	15	5	75	100	84	16	84	79.5
T60TJ	50	34	16	68	50	48	2	96	82
类一	5	5	0	100	999	997	2	99.8	99.9
类二	5	5	0	100	999	997	2	99.8	99.9
类三	2	2	0	100	999	999	0	100	100
类四	4	4	0	100	999	999	0	100	100
类五	1	1	0	100	999	999	0	100	100

表 7.5　IBLE 和 ID3 的平均预测正确率

类	IBLE/%	ID3/%
R	81.779	72.203
T60TJ	76.786	70.643
类二	98.334	89.322

对 IBLE 算法，在训练集中正、反例子数目做大的变化时，进行测试情况如表 7.6 所示。从表中可见，正例数不变化，反例数逐步减少时，正确识别率稍有提高。而反例数不变，正例数减少时，正确识别率显著下降。正、反例都下降时，正确识别率在逐步下降。

表 7.6　R 类例子数目变化时识别情况

训练集		对正例			对反例		
正例	反例	认对	认错	正确/%	认对	认错	正确/%
2363	2400	95	7	93.137	84	18	82.353
2363	1200	88	14	86.275	84	18	82.353
2363	400	91	11	89.216	99	3	97.059
2363	200	98	4	96.078	101	1	99.1
2363	100	98	4	98.078	101	1	99.1

续表

训练集		对正例			对反例		
正例	反例	认对	认错	正确/%	认对	认错	正确/%
2363	2400	95	7	93.137	84	18	82.353
1181	2400	76	26	74.51	71	31	69.608
393	2400	68	34	66.667	46	56	45.098
196	2400	54	48	52.941	35	67	34.314
98	2400	50	52	49.02	24	78	23.520
2363	2400	95	7	93.137	84	18	82.353
393	400	75	27	73.529	75	27	73.529
196	200	87	15	85.294	80	22	78.431
98	100	87	15	85.294	70	32	68.627

(2) 原因分析

IBLE 的预测正确率之所以比 ID3 高的原因在于：

- IBLE 用信道容量作为特征选择量，而 ID3 用互信息，信道容量不依赖于正、反例的比例，互信息依赖训练集中正反例的比例。
- ID3 在建树过程中，每次选择一个特征作为结点，不能较好地体现特征间的相关性。从几何角度来看，ID3 每次选择一个特征作为结点，在多维空间中等于每次利用一个与某坐标轴垂直的判定面，这样做不能充分利用训练集提供的信息。IBLE 在建树过程中每次循环选择多个特征构成规则，变量间的相关性得到较好的体现。从几何角度来看每次利用的判定面可以具有任意的方向，能较充分地利用训练集提供的信息。

(3) IBLE 决策规则树的特点

- IBLE 的决策规则树中的规则在表示和内容上与专家知识具有较高的一致性。以 R(苯)的决策规则树中第一条规则为例。规则列出了峰系列，与专家知识表示是一致的，第一条规则指出在 m/e=27、50～52、62～65、74～78、89～92、104～105 处应有峰。有关文献中认为含苯化合物的重要系列应是 m/e=38～39、50～52、63～65、75～78、91、105、119、113 等。比较一下知道，在列出的这 16 个峰中第一条规则包含了 12 个，而且都是权值较大的峰。专家知识中一般不指出哪些地方应无峰，而 IBLE 的规则中也指了出来，这是对专家知识的一种补充。而 ID3 的决策树在表示上与专家知识的相差较大，在内容上也不易做到与专家知识具有一致性(原因在于用互信息选择主要特征依赖于训练集中正、反例的比例，而实际问题中正、反例的比例不易确定)。
- 在训练集中，若正、反例数目变化较大，IBLE 得到的规则具有较好的稳定性。这在 R 的训练集中正、反例数目变化较大的情况下，IBLE 得出的各决策规则树中第一条规则，都含有相同的 41 个特征(m/e=41、42、43、50、51、54、55、56、57、58、59、62、63、64、65、67、68、69、70、71、72、75、76、77、78、81、82、83、84、85、89、90、91、92、96、97、98、100、104、105、143，包括有峰、无峰)，在相同的变化下 ID3 的决策树头两层 7 个

重要质量中,无共同的特征。

总之,IBLE 的规则与专家知识在内容上有较高的一致性,用 IBLE 获取的知识建立的专家系统对实例的判别进行解释时提供了良好的条件。这一点正是 ID3 的一个重要缺陷。显然,IBLE 比 ID3 优越。

4. 小结

我们提出的机器学习的信道模型,系统地论述了示例学习的信息论,利用新的特征选择量——信道容量,即用信道容量来选取重要特征的思想,不仅用于机器学习和数据挖掘之中,也可以用于模式识别的特征抽取。在上面的试验中,对 8 类化合物的质谱分类问题,用神经网络中的感知机和反向传播模型进行学习,由于特征太多,两种方法的效果都极差,利用信道容量进行特征提取后,再用感知机和 B-P 模型学习,都取得较好的效果。感知机的平均预测正确率为 79%,B-P 模型的平均预测正确率为 84%。文中提出的示例学习算法 IBLE 实现简单、学习正确性较高,所得知识在表示和内容上与专家知识有较高的一致性,而且特别适合于处理大规模的学习问题,可作为专家系统的知识获取工具。

习　　题

1. 信息论的基本原理是什么?
2. 学习信道模型是什么?
3. 为什么机器学习和数据挖掘可以利用信息论原理?
4. 自信息和互信息的含义是什么? 它们的计算公式是什么?
5. 信道容量的含义是什么? 它与互信息有什么关系?
6. 译码准则的基本思想是什么?
7. 决策树方法的基本思想是什么?
8. 说明 ID3 方法的建树算法步骤。
9. 设计用 ID3 决策树进行实例判别的判定算法。
10. 编制 ID3 算法的计算机程序,并用表 7.1 气候训练集例子进行测试。
11. 对于表 7.1 气候训练集,用 CLS 方法建树:任意选一字段项(如气温)为根结点,其字段项各取值为分支,对各分支数据子集重复上述操作,向下扩展此决策树,直到数据子集属于同一类数据(即叶结点)为止,并标记叶结点为 P 类或 N 类。

请比较 CLS 决策树与 ID3 决策树的优缺点。

12. 对表 7.1 气候训练集中,对“天气=晴”的数据子集,计算各特征(天气、气温、湿度、风)的互信息是多少? 哪个特征的互信息最大?
13. C4.5 方法对 ID3 方法的改进主要体现在什么地方?
14. 信息增益率与信息增益有什么不同? 在 C4.5 中为什么使用信息增益率作为分支标准?
15. 在 C4.5 中如何对连续属性进行处理?
16. IBLE 算法用什么来选择重要属性构造决策规则树结点?

17. IBLE 决策树的表示形式是什么？比较 IBLE 决策规则树和 ID3 决策树有什么不同？

18. IBLE 决策树中结点的表示形式是什么？

19. 设某例子集的 IBLE 决策规则树的结点规则为：

特征	a	b	c	d
权值	0.021	0.048	0.282	0.282
标准值	1	1	2	2

阈值　$S_n=0.564$　$S_p=0.585$

现有两个例子的特征取值分别为：

$a=1, b=2, c=2, d=2$

$a=1, b=1, c=1, d=2$

请用该结点规则判别它们属于{P 类、N 类、不能判别}中的哪种情况？

20. 说明 IBLE 决策规则树中结点中阈值 S_n 和 S_p 求解的思想？

21. 说明 IBLE 建规则算法。

22. 说明隐形眼镜简例中@3 类决策规则树的含义。

23. 说明从简例中@3 类决策规则树求出其相应的逻辑公式。

24. 请说明 IBLE 方法比 ID3 方法的技术进步点。

第8章 集合论方法

8.1 粗糙集方法

8.1.1 粗糙集概念

粗糙集(rough set)是波兰数学家Z. Pawlak于1982年提出的。粗糙集以等价关系(不可分辨关系)为基础,用于分类问题。它用上、下近似两个集合来逼近任意一个集合,该集合的边界线区域被定义为上近似集和下近似集之差集。上、下近似集可以通过等价关系给出确定的描述,边界域的元素数目可以被计算出来。而模糊集(fuzzy)是用隶属度来描述集合边界的不确定性,隶属度是人为给定的,不是计算出来的。

粗糙集理论用在数据库中的知识发现主要体现在:

(1) 利用等价关系对数据库进行属性约简。

(2) 利用集合的上、下近似关系获取分类规则。

1. 基本定义

(1) 信息表定义

信息表 $S=(U,R,V,f)$ 的定义如下。

U:是一个非空有限对象(元组)集合,$U=\{x_1,x_2,\cdots,x_n\}$,其中 x_i 为对象(元组)。

R:是对象的属性集合,分为两个不相交的子集,即条件属性 C 和决策属性 D,$R=C\cup D$。

V:是属性值的集合,V_a 是属性 $a\in R$ 的值域。

f:是 $U\times R\rightarrow V$ 的一个信息函数,它为每个对象 x 的每个属性 a 赋予一个属性值,即 $a\in R,x\in U,f_a(x)\in V_a$。

(2) 等价关系定义

对于 $\forall a\in A$(A 中包含一个或多个属性),$A\subset R,x\in U,y\in U$,它们的属性值相同,即

$$f_a(x)=f_a(y) \tag{8.1}$$

成立,称对象 x 和 y 是对属性 A 的等价关系,表示为

$$\mathrm{IND}(A)=\{(x,y)\mid(x,y)\in U\times U,\forall a\in A,f_a(x)=f_a(y)\} \tag{8.2}$$

(3) 等价类定义

在 U 中,对属性集 A 中具有相同等价关系的元素集合称为等价关系 IND(A)的等价类,表示为

$$[x]_A=\{y\mid(x,y)\in \mathrm{IND}(A)\} \tag{8.3}$$

(4) 划分的定义

在 U 中对属性 A 的所有等价类形成的划分表示为

$$A = \{E_i \mid E_i = [x]_A,\ i = 1,2\cdots\} \tag{8.4}$$

具有特性：

① $E_i \neq \Phi$。

② 当 $i \neq j$ 时，$E_i \cap E_j = \Phi$。

③ $U = \bigcup E_i$。

例 1 设 $U=\{a$(体温正常)$,b$(体温正常)$,c$(体温正常)$,d$(体温高)$,e$(体温高)$,f$(体温很高)$\}$，对于属性 A(体温)的等价关系有：

$$\text{IND}(A) = \{(a,b),(a,c),(b,c),(d,e),(e,d),(a,a),(b,b),(c,c),(d,d),(e,e),(f,f)\}$$

属性 A 的等价类有：

$$E_1 = [a]_A = [b]_A = [c]_A = \{a,b,c\}$$

$$E_2 = [d]_A = [e]_A = \{d,e\}$$

$$E_3 = [f]_A = \{f\}$$

U 中对属性 A 的划分为

$$A = \{E_1, E_2, E_3\} = \{\{a,b,c\},\{d,e\},\{f\}\}$$

2. 集合 X 的上、下近似关系

(1) 下近似定义

对任意一个子集 $X \subseteq U$，属性 A 的等价类 $E_i = [x]_A$，有

$$A_-(X) = \bigcup \{E_i \mid E_i \in A \wedge E_i \subseteq X\} \tag{8.5}$$

或

$$A_-(X) = \{x \mid [x]_A \subseteq X\} \tag{8.6}$$

表示等价类 $E_i=[x]_A$ 中的元素 x 都属于 X，即 $\forall x \in A_-(X)$，则 x 一定属于 X。$A_-(X)$表示下近似。

(2) 上近似定义

对任意一个子集 $X \subseteq U$，属性 A 的等价类 $E=[x]_A$，有

$$A^-(X) = \bigcup \{E_i \mid E_i \in A \wedge E_i \cap X \neq \Phi\} \tag{8.7}$$

或

$$A^-(X) = \{x \mid [x]_A \cap X \neq \Phi\} \tag{8.8}$$

表示等价类 $E_i=[x]_A$ 中的元素 x 可能属于 X，即 $\forall x \in A^-(X)$，则 x 可能属于 X，也可能不属于 X。$A^-(X)$表示上近似。

(3) 正域、负域和边界的定义

全集 U 可以划分为 3 个不相交的区域，即正域(Pos_A)、负域(NEG_A)和边界(BND_A)：

$$\text{Pos}_A(X) = A_-(X) \tag{8.9}$$

$$\text{NEG}_A(X) = U - A^-(X) \tag{8.10}$$

$$\text{BND}_A(X) = A^-(X) - A_-(X) \tag{8.11}$$

由此可见：

$$A^-(X) = A_-(X) + \text{BND}_A(X) \tag{8.12}$$

用图 8.1 说明正域、负域和边界，每一个小长方形表示一个等价类。

从图 8.1 中可以看出，任意一个元素 $x \in \mathrm{Pos}(X)$，一定属于 X；任意一个元素 $x \in \mathrm{NEG}(X)$，一定不属于 X；集合 X 的上近似是其正域和边界的并集，即

$$A^{-}(X) = \mathrm{Pos}_A(X) \cup \mathrm{BND}_A(X) \quad (8.13)$$

对于元素 $x \in \mathrm{BND}(X)$，是无法确定其是否属于 X，因此对任意元素 $x \in A^{-}(X)$，只知道 x 可能属于 X。

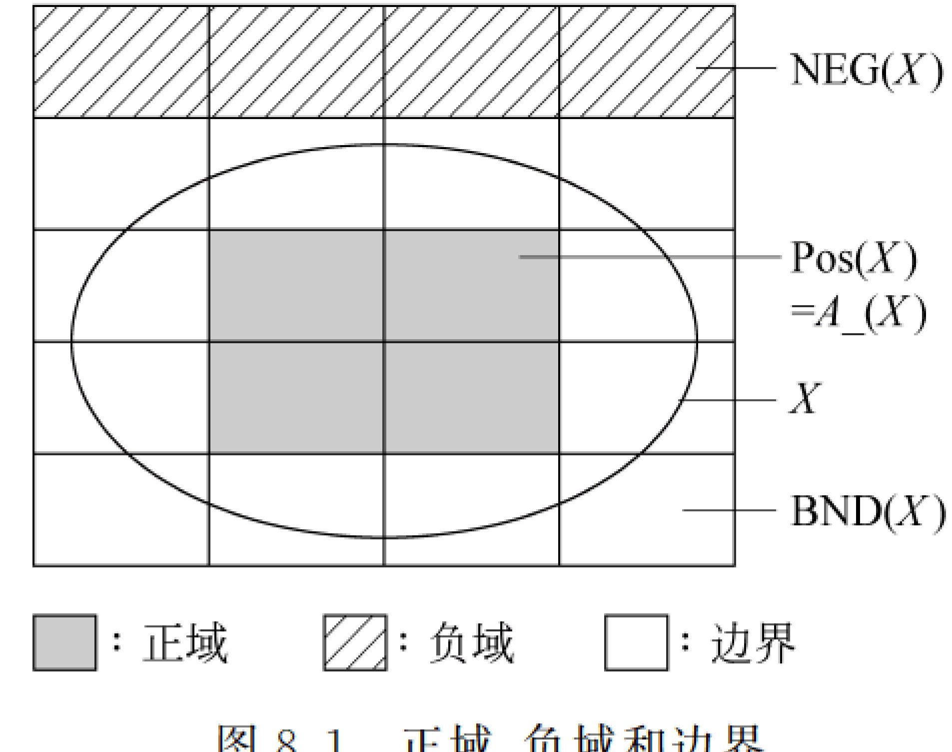


图 8.1　正域、负域和边界

(4) 粗糙集定义

若 $A^{-}(X)=A_{-}(X)$，即 $\mathrm{BND}(X)=\Phi$，即边界为空，称 X 为 A 的可定义集；否则 X 为 A 不可定义的，即 $A^{-}(X)\neq A_{-}(X)$，称 X 为 A 的 rough 集(粗糙集)。

(5) 确定度定义

$$\alpha_A(X) = \frac{|U| - |A^{-}X - A_{-}X|}{|U|} \tag{8.14}$$

其中 $|U|$ 和 $|A^{-}X-A_{-}X|$ 分别表示集合 U、$(A^{-}X-A_{-}X)$ 中的元素个数。

$\alpha_A(X)$ 的值反映了 U 中的能够根据 A 中各属性的属性值就能确定其属于或不属于 X 的比例，也即对 U 中的任意一个对象，根据 A 中各属性的属性值确定它属于或不属于 X 的可信度。

确定度性质：

$$0 \leqslant \alpha_A(X) \leqslant 1 \tag{8.15}$$

(1) 当 $\alpha_A(X)=1$ 时，U 中的全部对象能够根据 A 中各属性的属性值就可以确定其是否属于 X，X 为 A 的可定义集。

(2) 当 $0<\alpha_A(X)<1$ 时，U 中的部分对象根据 A 中各属性的属性值可以确定其是否属于 X，而另一部分对象是不能确定其是否属于 X。X 为 A 的部分可定义集。

(3) 当 $\alpha_A(X)=0$ 时，U 中的全部对象都不能根据 A 中各属性的属性值确定其是否属于 X，X 为 A 的完全不可定义集。

当 X 为 A 的部分可定义集或 X 为 A 的完全不可定义集时，称 X 为 A 的 rough 集(粗糙集)。

例 2　对例 1 的等价关系 A 有集合 $X=\{b,c,f\}$ 是粗糙集，计算集合 X 的下近似、上近似、正域、负域和边界。

U 中关于 A 的划分为

$$A = \{\{a,b,c\},\{d,e\},\{f\}\}$$

有：

$$X \cap \{a,b,c\} = \{b,e\} \neq \Phi$$
$$X \cap \{d,e\} = \Phi$$
$$X \cap \{f\} = \{f\} \neq \Phi$$

可知有：

$$A_-(X)=\{f\}$$
$$A^-(X)=\{a,b,c\}\cup\{f\}=\{a,b,c,f\}$$
$$\mathrm{Pos}_A(X)=A_-(X)=\{f\}$$
$$\mathrm{NEG}_A(X)=U-A^-(X)=\{d,e\}$$
$$\mathrm{BND}_A(X)=A^-(X)-A_-(X)=\{a,b,c\}$$

8.1.2 属性约简的粗糙集理论

1. 属性约简概念

在信息表中根据等价关系，可以用等价类中的一个对象(元组)来代表整个等价类，这实际上是按纵方向约简了信息表中的数据。对信息表中的数据按横方向进行约简就是看信息表中有无冗余的属性，即去除这些属性后能保持等价性，从而有相同的集合近似，使对象分类能力不会下降。约简后的属性集称做属性约简集，约简集通常不惟一，找到一个信息表的所有约简集不是一个在多项式时间里所解决的问题，求最小约简集(含属性个数最少的约简集)同样是一个困难问题，实际上它是一个 NP-hard 问题。因此研究者提出了很多启发式算法，如基于遗传算法的方法等。

(1) 约简定义

给定一个信息表 IT(U,A)，若有属性集 $B\subseteq A$，且满足 $\mathrm{IND}(B)=\mathrm{IND}(A)$，称 B 为 A 的一个约简，记为 $\mathrm{red}(A)$，即

$$B=\mathrm{red}(A) \tag{8.16}$$

(2) 核定义

属性集 A 的所有约简的交集称为 A 的核。记作

$$\mathrm{core}(A)=\bigcap\mathrm{red}(A) \tag{8.17}$$

$\mathrm{core}(A)$是 A 中为保证信息表中对象可精确定义的必要属性组成的集合，为 A 中不能约简的重要属性，是进行属性约简的基础。

上面的约简定义没有考虑决策属性，现研究条件属性 C 相对决策属性 D 的约简。

(3) 正域定义

设决策属性 D 的划分 $A=(y_1,y_2,\cdots,y_n)$，条件属性 C 相对于决策属性 D 的正域定义为

$$\mathrm{Pos}_C(D)=\bigcup C_-(y_j) \tag{8.18}$$

(4) 条件属性 C 相对于决策属性 D 的约简定义

若 $c\in C$，如果 $\mathrm{Pos}_{(C-\{c\})}(D)=\mathrm{Pos}_C(D)$，则称 c 是 C 中相对于 D 不必要的，即可约简的，否则称 c 是 C 中相对于 D 必要的。

(5) 条件属性 C 相对于决策属性 D 的核定义

若 $R\subseteq C$，如果 R 中每一个 $c\in R$ 都是相对于 D 必要的，则称 R 是相对于 D 独立的。如果 R 相对于 D 独立的，且 $\mathrm{Pos}_R(D)=\mathrm{Pos}_C(D)$，则称 R 是 C 中相对于 D 的约简，记为 $\mathrm{red}_D(C)$，所有这样约简的交称为 C 的 D 核，记为

$$\mathrm{core}_D(C)=\bigcap\mathrm{red}_D(C) \tag{8.19}$$

一般情况下，信息系统的属性约简集有多个，但约简集中属性个数最少的最有意义。

2. 属性约简实例

气候信息表是 4 个条件属性（天气 a_1，气温 a_2，湿度 a_3，风 a_4）和 1 个决策属性（类别 d），如表 8.1 所示。

表 8.1　气候信息表

No.	天气 a_1	气温 a_2	湿度 a_3	风 a_4	类别 d
1	晴	热	高	无风	N
2	晴	热	高	有风	N
3	多云	热	高	无风	P
4	雨	适中	高	无风	P
5	雨	冷	正常	无风	P
6	雨	冷	正常	有风	N
7	多云	冷	正常	有风	P
8	晴	适中	高	无风	N
9	晴	冷	正常	无风	P
10	雨	适中	正常	无风	P
11	晴	适中	正常	有风	P
12	多云	适中	高	有风	P
13	多云	热	正常	无风	P
14	雨	适中	高	有风	N

令 $C=\{a_1,a_2,a_3,a_4\}$，$D=\{d\}$

则：$\mathrm{IND}(C)=\{\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\},\{8\},\{9\},\{10\},\{11\},\{12\},\{13\},\{14\}\}$

$\mathrm{IND}(D)=\{\{1,2,6,8,14\},\{3,4,5,7,9,10,11,12,13\}\}$

$\mathrm{Pos}_C(D)=U$

(1) 计算缺少一个属性的等价关系：

$\mathrm{IND}(C\backslash\{a_1\})=\{\{1,3\},\{2\},\{4,8\},\{5,9\},\{6,7\},\{10\},\{11\},\{12,14\},\{13\}\}$

$\mathrm{IND}(C\backslash\{a_2\})=\{\{1,8\},\{2\},\{3\},\{4\},\{5,10\},\{6\},\{7\},\{9\},\{11\},\{12\},\{13\},\{14\}\}$

$\mathrm{IND}(C\backslash\{a_3\})=\{\{1\},\{2\},\{3,13\},\{4,10\},\{5\},\{6\},\{7\},\{8\},\{9\},\{11\},\{12\},\{13\},\{14\}\}$

$\mathrm{IND}(C\backslash\{a_4\})=\{\{1,2\},\{3\},\{4,14\},\{5,6\},\{7\},\{8\},\{9\},\{10\},\{11\},\{12\},\{13\}\}$

计算减少一个条件属性相对决策属性的正域：

$\mathrm{Pos}_{(C\backslash\{a_1\})}(D)=\{2,5,9,10,11,13\}\neq U$

$\mathrm{Pos}_{(C\backslash\{a_2\})}(D)=U=\mathrm{Pos}_C(D)$

$\mathrm{Pos}_{(C\backslash\{a_3\})}(D)=U=\mathrm{Pos}_C(D)$

$\mathrm{Pos}_{(C\backslash\{a_4\})}(D)=\{1,2,3,7,8,9,10,11,12,13\}\neq U$

由此可知，属性 a_2、a_3 是相对于决策属性 d 可省略的，但不一定可以同时省略，而属性 a_1 和 a_4 是相对决策属性不可省略的，因此：

$$\text{core}(C) = \{a_1, a_4\}$$

(2) 计算同时减少$\{a_2, a_3\}$的等价关系和正域：

$$\text{IND}(C \backslash \{a_2, a_3\}) = \{\{1,8,9\},\{2,11\},\{3,13\},\{4,5,10\},\{6,14\},\{7,12\}\}$$

$$\text{Pos}_{(C \backslash \{a_2, a_3\})}(D) = \{3,4,5,6,7,10,12,13,14\} \neq U$$

说明$\{a_2, a_3\}$同时是不可省略的。

(3) 在$\{a_2, a_3\}$中只能删除一个属性，即存在两个约简：

$$\text{red}_D(C) = \{\{a_1, a_2, a_3\},\{a_1, a_2, a_4\}\}$$

从实例计算可以看出，信息表的属性约简是在保持条件属性相对决策属性的分类能力不变的条件下，删除不必要的或不重要的属性。一般来讲，条件属性对于决策属性的相对约简不是惟一的，即可能存在多个相对约简。

3. 信息表的一致性

信息表中的对象(元组)x 按条件属性与决策属性关系看成一条决策规则，写成

$$\wedge f_{C_i}(x) \rightarrow f_d(x) \tag{8.20}$$

其中 C_i 表示多个条件属性，d 表示决策属性，$f_{C_i}(x)$表示对象 x 在属性 C_i 的取值，$\wedge$ 表示逻辑“与”关系。

(1) 一致性决策规则定义

如果对任一个对象 $y \neq x$，若条件属性有 $f_{C_i}(x) = f_{C_i}(y)$，则决策属性必须有

$$f_d(x) = f_d(y)$$

即一致性决策规则说明条件属性取值相同时，决策属性取值必须相同。

该定义允许：若条件属性有 $f_{C_i}(x) \neq f_{C_i}(y)$，则决策属性可以是 $f_d(x) = f_d(y)$或$f_d(x) \neq f_d(y)$。

(2) 信息表一致的定义

在信息表中如果所有对象的决策规则都是一致的，则该信息表是一致的，否则信息表是不一致的。

例如一个不一致信息表(见表 8.2)，属性集 $A = C \cup D$ 其中条件属性 $C = \{a, b, c\}$，决策属性 $D = \{d, e\}$。

表 8.2 不一致信息表

U	a	b	c	d	e
1	1	0	2	2	0
2	0	1	1	1	2
3	2	0	0	1	1
4	1	1	0	2	2
5	1	0	2	0	1
6	2	2	0	1	1
7	2	1	1	1	2
8	0	1	1	0	1

不一致信息表分解为一致信息表(见表 8.3)和完全不一致信息表(见表 8.4)。

表 8.3　一致信息表

U	a	b	c	d	e
3	2	0	0	1	1
4	1	1	0	2	2
6	2	2	0	1	1
7	2	1	1	1	2

表 8.4　完全不一致信息表

U	a	b	c	d	e
1	1	0	2	2	0
2	0	1	1	1	2
5	1	0	2	0	1
8	0	1	1	0	1

4. 保持信息表一致性的属性约简和属性值约简

信息表的简化一般有属性约简（约去不必要的属性）和属性值约简（消去一些无关紧要的属性值）。

（1）属性约简定义

在信息表中，将属性集中的属性逐个移去，每移去一个属性即检查其信息表，如果保持一致性，则该属性是可约去的。如果出现不一致则该属性不能被约去。不能约去的属性集合称为条件属性的核。

例如，有一致信息表 8.5。

表 8.5　一致信息表 1

U	a	b	c	d	e
1	1	0	2	1	1
2	2	1	0	1	0
3	2	1	2	0	2
4	1	2	2	1	1
5	1	2	0	0	2

在表 8.5 中移去属性 a 得表 8.6，它也是一致的。在表 8.5 中移去属性 b 得表 8.7，它也是一致的。

表 8.6　一致信息表 2

U	b	c	d	e
1	0	2	1	1
2	1	0	1	0
3	1	2	0	2
4	2	2	1	1
5	2	0	0	2

表 8.7　一致信息表 3

U	a	c	d	e
1	1	2	1	1
2	2	0	1	0
3	2	2	0	2
4	1	2	1	1
5	1	0	0	2

表 8.8　不一致信息表

U	a	b	d	e
1	1	0	1	1
2	2	1	1	0
3	2	1	0	2
4	1	2	1	1
5	1	2	0	2

在表 8.5 中移去属性 c，得表 8.8，它是不一致的。因为第 2 条规则 $a_2b_1 \rightarrow d_1e_0$ 和第 3 条规则 $a_2b_1 \rightarrow d_0e_2$ 是矛盾的。同样第 4 条规则和第 5 条规则也是不一致的。故属性 c 是不可约去的，它是属性集 $\{a, b, c\}$ 中的核，而 a 和 b 都是可被约去的，由此得到两个约简：

$$\text{red}_1(A) = \{a, c\} \quad 和 \quad \text{red}_2(A) = \{b, c\}$$

(2) 属性值约简命题

一条决策规则的条件属性值可消去，当且仅当消去后仍保持此规则的一致性。

例如，有信息表(见表 8.9)，其中 $U=\{1,2,3,4,5\}$，$C=\{a,b,c\}$，$D=\{d,e\}$。

表 8.9　信息表

U	a	b	c	d	e
1	1	0	2	1	1
2	2	1	0	2	0
3	2	1	2	0	2
4	1	2	2	1	1
5	1	2	0	0	2

对表 8.9 信息表的决策规则有：

① $a_1b_0c_2 \rightarrow d_1e_1$

② $a_2b_1c_0 \rightarrow d_2e_0$

③ $a_2b_1c_2 \rightarrow d_0e_2$

④ $a_1b_2c_2 \rightarrow d_1e_1$

⑤ $a_1b_2c_0 \rightarrow d_0e_2$

注：a_i 即 $a=i(i=1,2)$；b_j 即 $b=j(j=0,1,2)$；其他类同。

逐条检查规则，与其他条规则不存在条件属性值相同，故信息表是一致的。

对第一条规则 $a_1b_0c_2 \rightarrow d_1e_1$ 中消去 b_0 值(即取为 *)，与其他条规则中属性 b 的取值不

匹配，即 $a_1 * c_2 \to d_1 e_1$ 或 $a_1 c_2 \to d_1 e_1$，与其对应的规则为：

① $a_1 c_2 \to d_1 e_1$

② $a_2 c_0 \to d_2 e_0$

③ $a_2 c_2 \to d_0 e_2$

④ $a_1 c_2 \to d_1 e_1$

⑤ $a_1 c_0 \to d_0 e_2$

①规则和④规则的条件属性取值相同，决策属性取值也相同，保持一致，故该属性值可消去。

同样，对第一条规则 $a_1 b_0 c_2 \to d_1 e_1$ 消去 c_2 值，即 $a_1 b_0 * \to d_1 e_1$ 或 $a_1 b_0 \to d_1 e_1$；以及消去 a_1 值，即 $* b_0 c_2 \to d_1 e_1$ 或 $b_0 c_2 \to d_1 e_1$，均保持规则的一致性。可见这条规则①的核是空集，即 3 个属性值 a_1、b_0、c_2 均可被消去。

继续检查规则②$a_2 b_1 c_0 \to d_2 e_0$，它与 $a_2 c_0 \to d_2 e_0$（消去 b_1）和 $b_1 c_0 \to d_2 e_0$（消去 a_2）保持一致，而 $a_2 b_1 \to d_2 e_0$（消去 c_0）与③矛盾，所以属性 b_1 和 a_2 可消去。

同理，③、④ 和⑤ 中的 c_2 和 c_0 分别在其相应的规则中不能被消去，而其余在其相应的规则中均可被消去。经过如此属性值约简后，得到下面适合每条规则的核表，如表 8.10 所示。

表 8.10　仅包含决策规则核值

U	a	b	c	d	e
1	*	*	*	1	1
2	*	*	0	2	0
3	*	*	2	0	2
4	*	*	2	1	1
5	*	*	0	0	2

对表中每一条的 * 并不是全部消去而是可选消去，具体消去哪个 *，按如下命题处理。

（3）决策规则约简命题

属性集 C 中任意最小属性 a 的等价类 $[x]_a$ 的交集属于相应决策属性 D 的等价类 $[x]_D$，即

$$\bigcap [x]_a \subseteq [x]_D$$

则由此得到的最小条件属性 a 组成的条件相应决策属性的新决策规则是该条决策规则的约简。

例如，对表 8.9 参照表 8.10，求每一条决策规则的约简。

- 第一条规则的约简。其决策类 $[1]_{\{d,e\}} = \{1,4\}$；$[1]_a = \{1,4,5\}$；$[1]_c = \{1,3,4\}$。显然，$[1]_a \not\subset [1]_{\{d,e\}}$ 和 $[1]_c \not\subset [1]_{\{d,e\}}$，但 $[1]_b = \{1\} \subseteq [1]_{\{d,e\}}$ 和 $[1]_a \cap [1]_c = \{1,4\} \subseteq [1]_{\{d,e\}}$，所以得到两条约简的决策规则：

  1：$b_0 \to d_1 e_1$，　1′：$a_1 c_2 \to d_1 e_1$
- 第二条规则的约简。其决策类是 $[2]_{\{d,e\}} = \{2\}$；$[2]_a = \{2,3\}$；$[2]_b = \{2,3\}$；$[2]_c = \{2,5\}$，显然有：

$[2]_a \cap [2]_c = \{2\} \subseteq [2]_{\{d,e\}}$，$[2]_b \cap [2]_c = \{2\} \subseteq [2]_{\{d,e\}}$

得到两条约简规则：

2：$a_2 c_0 \rightarrow d_2 e_0$；　2′：$b_1 c_0 \rightarrow d_2 e_0$

同样可得，3、4、5 条规则的约简，它们分别为：

3：$a_2 c_2 \rightarrow d_0 e_2$；3′：$b_1 c_2 \rightarrow d_0 e_2$

4：$a_1 c_2 \rightarrow d_1 e_1$；4′：$b_2 c_2 \rightarrow d_1 e_1$

5：$a_1 c_0 \rightarrow d_0 e_2$；5′：$b_2 c_0 \rightarrow d_0 e_2$

所有约简的决策规则如表 8.11 所示。

表 8.11　包含所有约简决策规则

U	a	b	c	d	e
1	*	0	*	1	1
1′	1	*	2	1	1
2	2	*	0	2	0
2′	*	1	0	2	0
3	2	*	2	0	2
3′	*	1	2	0	2
4	1	*	2	1	1
4′	*	2	2	1	1
5	1	*	0	0	2
5′	*	2	0	0	2

注：1′和 4 规则相同，可以合并。

8.1.3　属性约简的粗糙集方法

1. 属性依赖度

(1) 属性依赖度定义

信息表中决策属性 D 依赖条件属性 C 的依赖度定义为

$$\gamma(C,D) = |\,\mathrm{Pos}_C(D)\,| \,/\, |\,U\,| \tag{8.21}$$

其中$|\mathrm{Pos}_C(D)|$表示正域 $\mathrm{Pos}_C(D)$的元素个数，$|U|$表示整个对象集合的个数。$\gamma(C,D)$的性质如下：

① 若 $\gamma=1$，意味着 $\mathrm{IND}(C)\subseteq\mathrm{IND}(D)$，即已知条件 C 下，可将 U 上全部个体准确分类到决策属性 D 的类别中去，即 D 完全依赖于 C。

② 若 $0<\gamma<1$，则称 D 部分依赖于 C(D rough 依赖于 C)，即在已知条件 C 下，只能将 U 上那些属于正域的个体分类到决策属性 D 的类别中去。

③ 若 $\gamma=0$，则称 D 完全不依赖 C，即利用条件 C 不能分类到 D 中的类别中去。

(2) 相关命题

根据属性依赖度定义，可以得到如下命题。

命题 1　如果依赖度 $\gamma=1$，则信息表是一致的，否则是不一致的。

命题 2　每个信息表都能惟一地分解成一个一致信息表($\gamma=1$)和一个完全不一致信息

表($\gamma=0$)。

2. 属性重要度

(1) 属性重要度定义

$C,D\subset A$,C 为条件属性集,D 为决策属性集,$a\in C$,属性 a 关于 D 的重要度定义为

$$\mathrm{SGF}(a,C,D)=\gamma(C,D)-\gamma(C-\{a\},D) \tag{8.22}$$

其中 $\gamma(C-\{a\},D)$表示在 C 中缺少属性 a 后,条件属性与决策属性的依赖程度。$\mathrm{SGF}(a,C,D)$表示 C 中缺少属性 a 后,导致不能被准确分类的对象在系统中所占的比例。

(2) $\mathrm{SGF}(a,C,D)$的性质

① $\mathrm{SGF}(a,C,D)\in[0,1]$。

② 若 $\mathrm{SGF}(a,C,D)=0$,表示属性 a 关于 D 是可省的。因为从属性集中去除属性 a 后,$C-\{a\}$中的信息仍能准确划分到各决策类中去。

③ $\mathrm{SGF}(a,C,D)\neq 0$,表示属性 a 关于 D 是不可省的。因为从属性集 C 中去除属性 a 后,某些原来可被准确分类的对象不再被准确划分。

3. 最小属性集概念

对信息系统的最广泛应用是数据库。在数据库中根据决策属性将一组对象划分为各不相交的等价集(决策类),希望能通过条件属性来决定每一个决策类,并产生每一个类的判定规则。大多数情况下,对每个给定的学习任务,数据库中存在一些不重要属性,希望找到一个最小的相关属性集,具有与全部条件属性同样的区分决策属性所划分的决策类的能力,从最小属性集中产生的规则会更简练和更有意义。

最小属性集定义:设 C、D 分别是信息系统 S 的条件属性集和决策属性集,属性集 $P(P\subseteq C)$是 C 的一个最小属性集,当且仅当 $\gamma(P,D)=\gamma(C,D)$并且 $\forall P'\subset P,\gamma(P',D)\neq\gamma(P,D)$,说明若 P 是C 的最小属性集,则 P 具有与C 同样的区分决策类的能力。

需要注意的是,C 的最小属性集一般是不惟一的,而要找到所有的最小属性集是一个 NP 问题。在大多数应用中,没有必要找到所有的最小属性集。用户可以根据不同的原则来选择一个认为最好的最小属性集。比如,选择具有最少属性个数的最小属性集。

8.1.4 粗糙集方法的规则获取

通过分析 U 中的两个划分 $C=\{E_i\}$和 $D=\{Y_j\}$之间的关系,把 C 视为分类条件,D 视为分类结论,可以得到下面的分类规则:

(1) 当 $E\cap Y_j\neq\varnothing$时,则有:

$$r_{ij}:\mathrm{Des}(E_i)\rightarrow\mathrm{Des}(Y_j) \tag{8.23}$$

$\mathrm{Des}(E_i)$和 $\mathrm{Des}(Y_j)$分别是等价集 E_i和等价集 Y_j中的特征描述。

① 当 $E\cap Y_j=E_i$时(E_i完全被 Y_j包含),即下近似,建立的规则 r_{ij}是确定的,规则的可信度 $cf=1.0$。

② 当 $E\cap Y_j\neq E_i$时(E_i部分被 Y_j包含),即上近似,建立的规则 r_{ij}是不确定的,规则的可信度为

$$cf = \frac{|E_i \cap Y_j|}{|E_i|} \tag{8.24}$$

用图 8.2 表示 E_i 和 Y_j 的上、下近似关系。

(2) 当 $E_i \cap Y_j = \varnothing$ 时(E_i 不被 Y_j 包含),E_i 和 Y_j 不能建立规则。

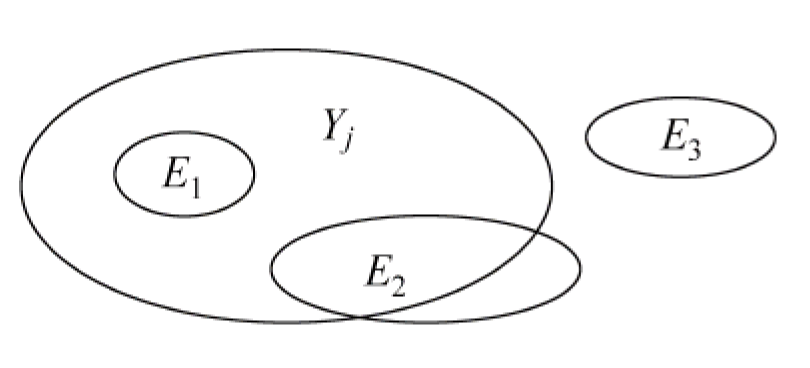


图 8.2 E_i 和 Y_j 的上、下近似关系

8.1.5 粗糙集方法的应用实例

通过实例说明属性约简和规则获取方法,见表 8.12 的数据。

表 8.12 流感实例数据

项目	C(条件属性)			D(决策属性)
U	头痛(a)	肌肉痛(b)	体温(c)	流感(d)
e_1	是(1)	是(1)	正常(0)	否(0)
e_2	是(1)	是(1)	高(1)	是(1)
e_3	是(1)	是(1)	很高(2)	是(1)
e_4	否(0)	是(1)	正常(0)	否(0)
e_5	否(0)	否(0)	高(1)	否(0)
e_6	否(0)	是(1)	很高(2)	是(1)
e_7	是(1)	否(0)	高(1)	是(1)

1. 等价集下近似和依赖度的计算

(1) 条件属性 $C(a,b,c)$ 的等价集

由于各元组(对象)之间不存在等价关系,每个元组组成一个等价集,共 7 个:

$E_1\{e_1\}, E_2\{e_2\}, E_3\{e_3\}, E_4\{e_4\}, E_5\{e_5\}, E_6\{e_6\}, E_7\{e_7\}$。

(2) 决策属性 $D(d)$ 的等价集

按属性取值,共有两个等价集:Y_1:$\{e_1, e_4, e_5\}$;Y_2:$\{e_2, e_3, e_6, e_7\}$。

(3) 决策属性的各等价集的下近似集

$C_Y_1 = \{E_1, E_4, E_5\} = \{e_1, e_4, e_5\}$

$C_Y_2 = \{E_2, E_3, E_6, E_7\} = \{e_2, e_3, e_6, e_7\}$

此例不存在上近似集。

(4) 计算 $\mathrm{Pos}(C,D)$ 和 $\gamma(C,D)$

$\mathrm{Pos}(C,D) = C_Y_1 \cup C_Y_2 = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$

$|\mathrm{Pos}(C,D)| = 7, |U| = 7, \gamma(C,D) = 1$

2. 各属性重要度计算

(1) a 的重要度计算

- 条件属性 $C(b,c)$ 的等价集:

$$E_1\{e_1, e_4\}, E_2\{e_2\}, E_3\{e_3, e_6\}, E_4\{e_5, e_7\}$$

- 决策属性 $D(d)$ 的等价集仍为 Y_1 和 Y_2。

- 决策属性的各等价集的下近似集：

$$C_Y_1 = \{E_1\} = \{e_1, e_4\}$$

$$C_Y_2 = \{E_2, E_3\} = \{e_2, e_3, e_6\}$$

- 计算 $\text{Pos}(C-\{a\}, D)$和 $\gamma(C-\{a\}, D)$：

$$\text{Pos}(C-\{a\}, D) = C_Y_1 \cup C_Y_2 = \{e_1, e_2, e_3, e_4, e_6\}$$

$$|\text{Pos}(C-\{a\}, D)| = 5$$

$$\gamma(C-\{a\}, D) = 5/7$$

- 属性 a 的重要程度：

$$\text{SGF}(C-\{a\}, D) = \gamma(C, D) - \gamma(C-\{a\}, D) = 2/7 \neq 0$$

- 结论：属性 a 是不可省略的。

(2) b 的重要度计算

- 条件属性 $C(a,c)$的等价集：去掉属性 b 后，元组中只出现 e_2 和 e_7 的等价，其他元组均不等价，等价集共 6 个：

$$E_1\{e_1\}, E_2\{e_2, e_7\}, E_3\{e_3\}, E_4\{e_4\}, E_5\{e_5\}, E_6\{e_6\}$$

- 决策属性 $D(d)$的等价集仍为 Y_1 和 Y_2。
- 决策属性的各等价集的下近似集：

$$C_Y_1 = \{E_1, E_4, E_5\} = (e_1, e_4, e_5)$$

$$C_Y_2 = \{E_2, E_3, E_6\} = (e_2, e_7, e_3, e_6)$$

- 计算 $\text{Pos}(C-\{b\}, D)$：

$$\text{Pos}(C-\{b\}, D) = C_Y_1 \cup C_Y_2 = (e_1, e_2, e_3, e_4, e_5, e_6, e_7)$$

$$|\text{Pos}(C-\{b\}, D)| = 7, \gamma(C-\{a\}, D) = 1$$

- 属性 b 的重要度：

$$\text{SGF}(C-\{b\}, D) = \gamma(C, D) - \gamma(C-\{a\}, D) = 0$$

- 结论：属性 b 是可省略的。

3. 简化数据表

在原数据表中删除肌肉痛(b)属性后，元组 e_7 和 e_2 相同，合并成表 8.13 所示的简化数据表。

表 8.13　流感数据简化表

U	头痛(a)	体温(c)	流感(d)
e_1'	是(1)	正常(0)	否(0)
e_2'	是(1)	高(1)	是(1)
e_3'	是(1)	很高(2)	是(1)
e_4'	否(0)	正常(0)	否(0)
e_5'	否(0)	高(1)	否(0)
e_6'	否(0)	很高(2)	是(1)

4. 等价集、上下近似集的计算

(1) 条件属性的等价集

由于各元组之间不存在等价关系，故有 6 个等价集：$E_1'\{e_1'\}$，$E_2'\{e_2'\}$，$E_3'\{e_3'\}$，$E_4'\{e_4'\}$，$E_5'\{e_5'\}$，$E_6'\{e_6'\}$。

(2) 决策属性 $D(d)$的等价集

按属性取值，共有两个等价集：$Y_1'\{e_1',e_4',e_5'\}$和 $Y_2'\{e_2',e_3',e_6'\}$。

5. 获取规则

图 8.3 是 Y_1'与 E_1'、E_4'、E_5'最小包含图。

(1) 由于 $E_1'\cap Y_1'=E_1'$，$E_4'\cap Y_1'=E_4'$，$E_5'\cap Y_1'=E_5'$，有规则

r_{11}：$\text{Des}(E'_1)\to\text{Des}(Y_1')$，即 $a=1\wedge c=0\to d=0$，$cf=1$。

r_{41}：$\text{Des}(E_4')\to\text{Des}(Y_1')$，即 $a=0\wedge c=0\to d=0$，$cf=1$。

r_{51}：$\text{Des}(E_5')\to\text{Des}(Y_1')$，即 $a=0\wedge c=1\to d=0$，$cf=1$。

(2) 由于 $E_2'\cap Y_2'=E_2'$，$E_3'\cap Y_2'=E_3'$，$E_6'\cap Y_2'=E_6'$，有规则

r_{22}：$\text{Des}(E_2')\to\text{Des}(Y_2')$，即 $a=1\wedge c=1\to d=1$，$cf=1$。

r_{32}：$\text{Des}(E_3')\to\text{Des}(Y_2')$，即 $a=1\wedge c=2\to d=1$，$cf=1$。

r_{62}：$\text{Des}(E_6')\to\text{Des}(Y_2')$，即 $a=0\wedge c=2\to d=1$，$cf=1$。

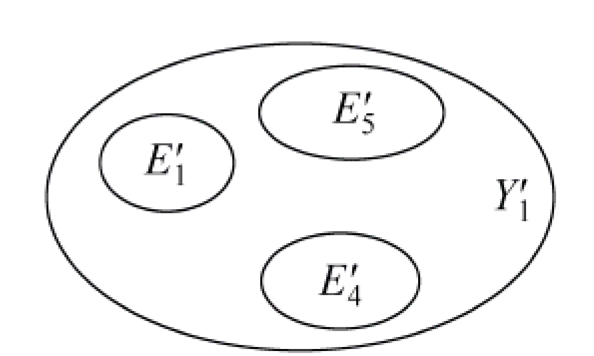


图 8.3 Y_1'与 E_1'、E_4'、E_5'最小包含图

6. 规则化简

(1) 对 r_{11} 和 r_{41}进行合并，有

$$(a=0\vee a=1)\wedge c=0\to d=0$$

其中 a 的取值包括了全部取值，故属性 a 可删除，即

$$c=0\to d=0$$

(2) 对 r_{32} 和 r_{62}进行合并，有

$$(a=1\vee a=0)\wedge c=2\to d=1$$

同样，可删除属性 a，得到

$$c=2\to d=1$$

7. 最后的规则

(1) 体温=正常→流感=否(即 $c=0\to d=0$)。

(2) 头痛=否∧体温=高→流感=否(即 $a=0\wedge c=1\to d=0$)。

(3) 体温=很高→流感=是(即 $c=2\to d=1$)。

(4) 头痛=是∧体温=高→流感=是(即 $a=1\wedge c=1\to d=1$)。

8.2 关联规则挖掘

关联规则(association rule)挖掘是发现大量数据库中项集之间的关联关系。随着大量数据的增加和存储，许多人士对于从数据库中挖掘关联规则越来越感兴趣。从大量商业事

务中发现有趣的关联关系，可以帮助许多商业决策的制定，如分类设计、交叉购物等。

目前，关联规则挖掘已经成为数据挖掘领域重要的研究方向。关联规则模式属于描述型模式，发现关联规则的算法属于无监督学习的方法。

Agrawal 等人于 1993 年首先提出了挖掘顾客交易数据库中项集间的关联规则问题，以后诸多的研究人员对关联规则的挖掘问题进行了大量的研究。他们的工作包括对原有的算法进行优化，如引入随机采样、并行的思想等，以提高算法挖掘规则的效率，并对关联规则的应用进行推广。

最近也有独立于 Agrawal 的频繁集方法的工作，以克服频繁集方法的一些缺陷，探索挖掘关联规则的新方法。同时随着 OLAP 技术的成熟和应用，将 OLAP 和关联规则结合也成了一个重要的方向。也有一些工作注重于对挖掘到的模式的价值进行评估，他们提出的模型建议了一些值得考虑的研究方向。

本章主要给出了关联规则挖掘的基本概念和核心挖掘算法。

8.2.1 关联规则的挖掘原理

关联规则是发现交易数据库中不同商品（项）之间的联系，这些规则找出顾客购买行为模式，如购买了某一商品对购买其他商品的影响。发现这样的规则可以应用于商品货架设计、货存安排以及根据购买模式对用户进行分类。现实中，这样的例子很多。最典型的，例如超级市场利用前端收款机收集存储了大量的售货数据，这些数据是一条条的购买事务记录，每条记录存储了事务处理时间，顾客购买的物品、物品的数量及金额等。这些数据中常常隐含如下形式的关联规则：

在购买铁锤的顾客当中，有 70％的人同时购买了铁钉。

这些关联规则很有价值，商场管理人员可以根据这些关联规则更好地规划商场，如把铁锤和铁钉这样的商品摆放在一起，能够促进销售。

有些数据不像售货数据那样很容易就能看出一个事务是许多物品的集合，但稍微转换一下思考角度，仍然可以像售货数据一样处理。比如人寿保险，一份保单就是一个事务。保险公司在接受保险前，往往需要记录投保人详尽的信息，有时还要到医院做身体检查。保单上记录有投保人的年龄、性别、健康状况、工作单位、工作地址、工资水平等。

这些投保人的个人信息就可以看作事务中的物品。通过分析这些数据，可以得到类似以下的关联规则：

年龄在 40 岁以上，工作在 A 区的投保人当中，有 45％的人曾经向保险公司索赔过。在这条规则中，“年龄在 40 岁以上”是物品甲，“工作在 A 区”是物品乙，“向保险公司索赔过”则是物品丙。可以看出来，A 区可能污染比较严重，环境比较差，导致工作在该区的人健康状况不好，索赔率也相对比较高。

1. 基本原理

设 $I=\{i_1, i_2, \cdots, i_m\}$ 是项（Item）的集合。记 D 为事务（Transaction）的集合（事务数据库），事务 T 是项的集合，并且 $T \subseteq I$。对每一个事务有惟一的标识，如事务号，记作 TID。设 A 是 I 中一个项集，如果 $A \subseteq T$，那么称事务 T 包含 A。

定义 1 关联规则是形如 $A \to B$ 的蕴涵式，这里 $A \subset I, B \subset I$，并且 $A \cap B = \Phi$。

定义 2 规则的支持度。规则 $A \to B$ 在数据库 D 中具有支持度 S，表示 S 是 D 中事务同时包含 AB 的百分比，它是概率 $P(AB)$，即

$$S(A \to B) = P(AB) = \frac{|AB|}{|D|} \tag{8.25}$$

其中 $|D|$ 表示事务数据库 D 的个数，$|AB|$ 表示 A、B 两个项集同时发生的事务个数。

定义 3 规则的可信度。

规则 $A \to B$ 具有可信度 C，表示 C 是包含 A 项集的同时也包含 B 项集，相对于包含 A 项集的百分比，这是条件概率 $P(B|A)$，即

$$C(A \to B) = P(B \mid A) = \frac{|AB|}{|A|} \tag{8.26}$$

其中 $|A|$ 表示数据库中包含项集 A 的事务个数。

定义 4 阈值。为了在事务数据库中找出有用的关联规则，需要由用户确定两个阈值：最小支持度(min_sup)和最小可信度(min_conf)。

定义 5 项的集合称为项集(Itemset)，包含 k 个项的项集称为 k-项集。如果项集满足最小支持度，则称为频繁项集(frequent itemset)。

定义 6 关联规则。同时满足最小支持度(min_sup)和最小可信度(min_conf)的规则称为关联规则，即 $S(A \to B) > \text{min_sup}$ 且 $C(A \to B) > \text{min_conf}$ 成立时，规则 $A \to B$ 称为关联规则，也可以称为强关联规则。

2. 关联规则挖掘过程

关联规则的挖掘一般分为如下两个过程。

(1) 找出所有的频繁项集：根据定义，这些项集的频繁性至少和预定义的最小支持数目一样。

(2) 由频繁项目产生关联规则：根据定义，这些规则必须满足最小支持度和最小可信度。

这两步中，第二步是在第一步的基础上进行的，工作量非常小。挖掘关联规则的总体性能由第一步决定。

3. 关联规则的兴趣度

关联规则主要是考虑同时购买商品的事务的相关性。对于不购买商品的事务与购买商品的事务的关系的研究，需要引入兴趣度概念。

先通过一个具体的例子来说明不购买商品与购买商品的关系。设 $I=$(咖啡，牛奶)，交易集 D，经过对 D 的分析，得到如表 8.14 所示的表格。

表 8.14 交易集的分析

项目	买咖啡	不买咖啡	合计
买牛奶	20	5	25
不买牛奶	70	5	75
合计	90	10	100

由表 8.14 可以了解到，如果设定 min_sup=0.2，min_conf=0.8，按照现有的挖掘算法可以得到如下的关联规则：

$$\text{买牛奶} \rightarrow \text{买咖啡}, \quad S = 0.2, \quad C = 0.8 \tag{8.27}$$

即 80%的人买了牛奶就会买咖啡。这一点从逻辑上是完全合理正确的。

但从表 8.14 中同时也可以毫不费神地得到结论：90%的人肯定会买咖啡。换句话说，买牛奶这个事件对于买咖啡这个事件的刺激作用(80%)并没有想象中的(90%)那么大。反而是规则

$$\text{买咖啡} \rightarrow \text{不买牛奶}, \quad S = 0.7, \quad C = 0.78 \tag{8.28}$$

的支持度和可信度分别为 0.7 和 0.78，更具有商业销售的指导意义。

从上面这个例子可以发现，目前基于支持度-可信度的关联规则的评估体系存在着问题；同时，现有的挖掘算法只能挖掘出类似于式(8.27)的规则，而对于类似于式(8.28)的带有类似于“不买牛奶”之类的负属性项的规则却无能为力，而这种知识往往具有更重要的价值。国内外围绕这个问题展开了许多研究。引入兴趣度概念，分析项集 A 与项集 B 的关系程度。

定义 7 兴趣度为

$$I(A \rightarrow B) = \frac{P(AB)}{P(A)P(B)} \tag{8.29}$$

式(8.29)反映了项集 A 与项集 B 的相关程度。若

$$I(A \rightarrow B) = 1, \quad \text{即 } P(AB) = P(A)P(B)$$

表示项集 A 出现和项集 B 出现是相互独立的。若

$$I(A \rightarrow B) < 1$$

表示 A 出现和 B 出现是负相关的。若

$$I(A \rightarrow B) > 1$$

表示 A 出现和 B 出现是正相关的。意味着 A 的出现蕴涵 B 的出现。

在兴趣度的使用中，一条规则的兴趣度越大于 1，说明对这条规则越感兴趣(即其实际利用价值越大)；一条规则的兴趣度越小于 1，说明对这条规则的反面规则越感兴趣(即其反面规则的实际利用价值越大)；显然，兴趣度 I 不小于 0。

下面从兴趣度的角度来看一下前面那个牛奶与咖啡的例子，列出所有可能的规则描述及其对应的支持度、可信度和兴趣度，如表 8.15 所示。

表 8.15 所有可能的关联规则

项目	rules	S	C	I
1	买牛奶→买咖啡	0.2	0.8	0.89
2	买咖啡→买牛奶	0.2	0.22	0.89
3	买牛奶→不买咖啡	0.05	0.2	2
4	不买咖啡→买牛奶	0.05	0.5	2
5	不买牛奶→买咖啡	0.7	0.93	1.037
6	买咖啡→不买牛奶	0.7	0.78	1.037
7	不买牛奶→不买咖啡	0.05	0.067	0.67
8	不买咖啡→不买牛奶	0.05	0.2	0.87

在此只考虑第1、2、3、6共4条规则。由于 $I_1, I_2 < 1$，所以在实际中它的价值不大；$I_3, I_6 > 1$，都可以列入进一步考虑的范围。

公式(8.29)等价于

$$I(A \rightarrow B) = \frac{P(AB)}{P(A)P(B)} = \frac{P(B \mid A)}{P(B)} \tag{8.30}$$

公式(8.30)，有人称之为作用度(lift)，表示关联规则 $A \rightarrow B$ 的“提升”。如果作用度(兴趣度)不大于1，则此关联规则就没有意义了。

概括地说，可信度是对关联规则的准确度的衡量。支持度是对关联规则重要性的衡量。支持度说明了这条规则在所有事务中有多大的代表性，显然支持度越大，关联规则越重要。有些关联规则可信度虽然很高，但支持度却很低，说明该关联规则实用的机会很小，因此也不重要。

兴趣度(作用度)描述了项集 A 对项集 B 的影响力的大小。兴趣度(作用度)越大，说明项集 B 受项集 A 的影响越大。

8.2.2 Apriori 算法的基本思想

Agrawal 等人于1993年首先提出了挖掘顾客交易数据库中项集间的关联规则问题，设计了基于频繁集理论的 Apriori 算法。以后诸多的研究人员对关联规则的挖掘问题进行了大量的研究。他们的工作包括对原有的算法进行优化，如引入随机采样、并行的思想等，以提高算法挖掘规则的效率；提出各种变体，如泛化的关联规则、周期关联规则等，对关联规则的应用进行推广。

Apriori 是挖掘关联规则的一个重要方法。这是一个基于两阶段频繁集思想的方法，将关联规则挖掘算法的设计分解为两个子问题：

- 找到所有支持度大于最小支持度的项集(itemset)，这些项集称为频繁集(frequent itemset)。
- 使用第一个子问题找到的频繁集产生期望的规则。

Apriori 使用一种称做逐层搜索的迭代方法，即“K-项集”用于探索“$K+1$-项集”。首先，找出频繁“1-项集”的集合。该集合记作 L_1。L_1 用于找频繁“2-项集”的集合 L_2，而 L_2 用于找 L_3，如此下去，直到不能找到“K-项集”。找每个 L_K 需要一次数据库扫描。

1. Apriori 性质

性质 频繁项集的所有非空子集都必须也是频繁的。

该性质表明，如果项集 B 不满足最小支持度阈值 min_sup，则 B 不是频繁的，即 $P(B) <$ min_sup。如果项集 A 添加到 B，则结果项集(即 $B \cup A$)不可能比 B 更频繁出现。因此，$B \cup A$ 也不是频繁的，即 $P(B \cup A) <$ min_sup。

Apriori 性质可用于压缩搜索空间。

2. “K-项集”产生“$K+1$-项集”

设 K-项集 L_K，$K+1$ 项集 L_{K+1}，产生 L_{K+1} 的候选集 C_{K+1}。有公式：

$$C_{K+1} = L_K \times L_K = \{X \cup Y, \text{其中 } X, Y \in L_K, \mid XY \mid = K+1\}$$

其中 C_1是 1-项集的集合,取自所有事务中的单项元素。如:

$$L_1 = \{\{A\},\{B\}\}$$

$$C_2 = \{A\} \cup \{B\} = \{A,B\}, \text{且} \mid AB \mid = 2$$

$$L_2 = \{\{A,B\},\{A,C\}\}$$

$$C_3 = \{A,B\} \cup \{A,C\} = \{A,B,C\}, \text{且} \mid ABC \mid = 3$$

3. Apriori 算法中候选项集与频繁项集的产生实例

有如表 8.16 所列的事务数据库,Apriori 算法步骤如下:对于下述一个例子事务数据库产生频繁项集。

表 8.16　事务数据库例

事务 ID	事务的项目集
T_1	A,B,E
T_2	B,D
T_3	B,C
T_4	A,B,D
T_5	A,C
T_6	B,C
T_7	A,C
T_8	A,B,C,E
T_9	A,B,C

(1) 在算法的第一次迭代,每个项都是候选 1-项集的集合 C_1的成员。算法扫描所有的事务,对每个项的出现次数计数,如图 8.4 中的第 1 列。

(2) 假定最小事务支持计数为 2(即 min_sup=2/9≈22%),可以确定频繁 1-项集的集合 L_1。它由具有最小支持度的候选 1-项集组成,如图 8.4 中的第 2 列。

(3) 为发现频繁 2-项集的集合 L_2,算法使用 $L_1 * L_1$来产生候选集 C_2,如图 8.4 中的第 3 列。

(4) 扫描 D 中事务,计算 C_2中每个候选项集的支持度计数,如图 8.4 中的第 4 列。

(5) 确定频繁 2-项集的集合 L_2,它由具有最小支持度的 C_2中的候选 2-项集组成,如图 8.4 的第 5 列。

(6) 候选 3-项集的集合 C_3的产生,仍按(3)进行。得到候选集:

$$C_3 = \{\{A,B,C\},\{A,B,E\},\{A,C,E\},\{B,C,D\},\{B,C,E\},\{B,D,E\}\}$$

按 Apriori 性质,频繁项集的所有子集必须是频繁的。由于$\{A,D\}$,$\{C,D\}$,$\{C,E\}$,$\{D,E\}$不是频繁项集,故 C_3中后 4 个候选不可能是频繁的,在 C_3中删除它们,如图 8.4 中第 6 列。

扫描 D 中事务,对 C_3中的候选项集计算支持度计数,见图 8.4 第 7 列。

(7) 确定 L_3,它由具有最小支持度的 C_3中候选 3-项集组成,如图 8.4 中第 8 列。

(8) 按公式产生候选 4-项集的集合 C_4,产生结果$\{A,B,C,E\}$,这个项集被剪去,因为它

的子集$\{B,C,E\}$不是频繁的。这样$L_4=\Phi$。此算法终止。L_3是最大的频繁项集，即$\{A,B,C\}$和$\{A,B,E\}$。

具体产生过程如图8.4所示。

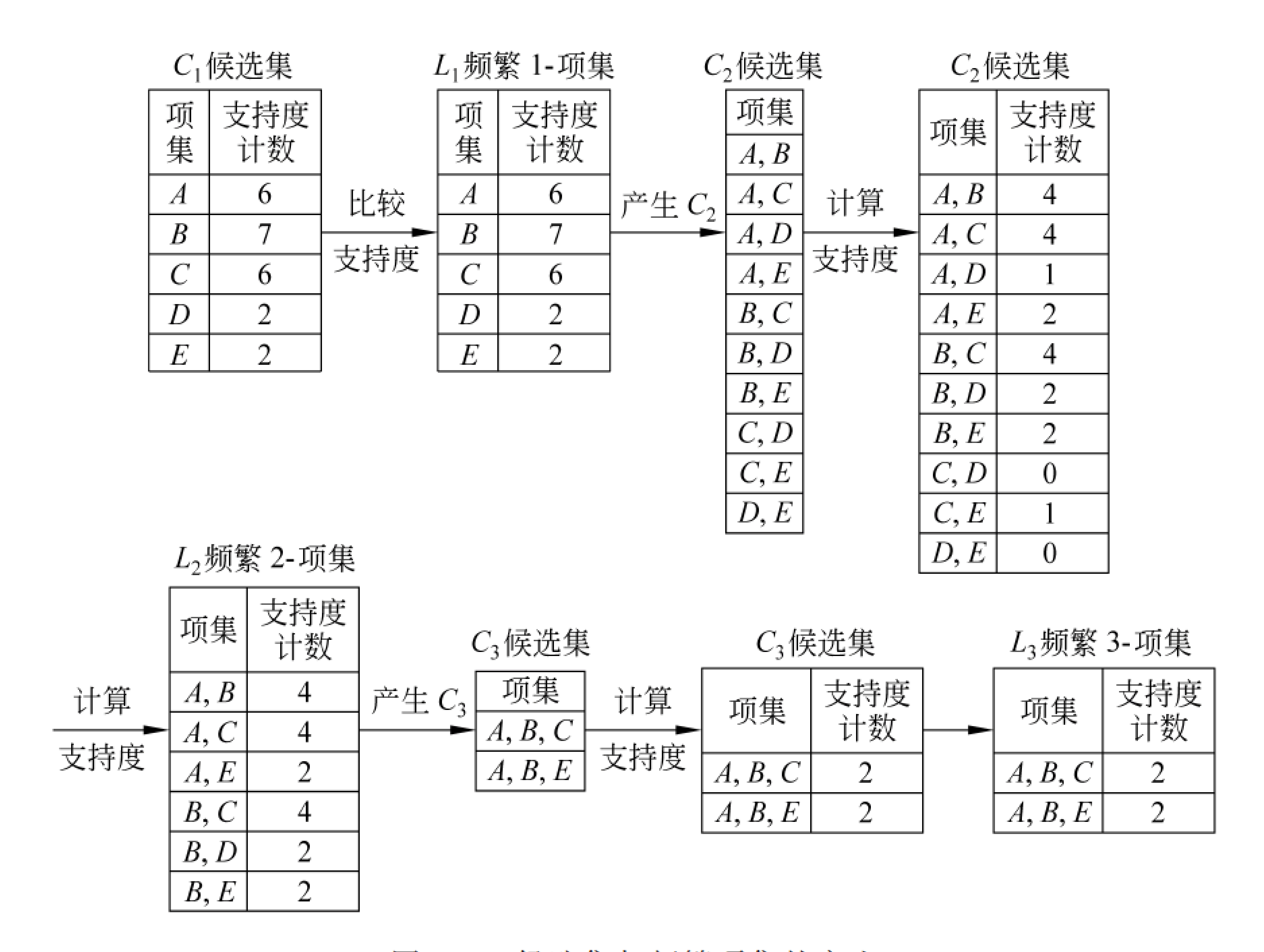


图8.4　候选集与频繁项集的产生

4. 产生关联规则

由频繁项集产生关联规则的工作相对简单一点。根据前面提到的置信度的定义，关联规则的产生如下：

(1) 对于每个频繁项集L，产生L的所有非空子集；

(2) 对于L的每个非空子集S，如果$\frac{|L|}{|S|}\geqslant \text{min_conf}$，则输出规则：$S\rightarrow L-S$。

注：$L-S$表示在项集L中除去S子集的项集。$|L|$和$|S|$表示项集L和S的计数。

由于规则由频繁项目集产生，每个规则都自动满足最小支持度。

在表8.16事务数据库中，频繁项集$L=\{A,B,E\}$可以由L产生哪些关联规则？L的非空子集S有：$\{A,B\}$，$\{A,E\}$，$\{B,E\}$，$\{A\}$，$\{B\}$，$\{E\}$。可得到关联规则如下：

$A\wedge B\rightarrow E$　　confidence=2/4=50%

$A\wedge E\rightarrow B$　　confidence=2/2=100%

$B\wedge E\rightarrow A$　　confidence=2/2==100%

$A\rightarrow B\wedge E$　　confidence=2/6≈33%

$B\rightarrow A\wedge E$　　confidence=2/7≈29%

$E\rightarrow A\wedge B$　　confidence=2/2=100%

假设最小可信度为60%，则最终输出的关联规则为：

$A \wedge E \rightarrow B$　　confidence=100%

$B \wedge E \rightarrow A$　　confidence=100%

$E \rightarrow A \wedge B$　　confidence=100%

对于频繁项集{A,B,C},同样可得其他关联规则。

8.2.3　Apriori 算法程序

为了生成所有频繁集,使用了递推的方法。程序包括 apriori_gen 子程序产生候选,完成连接和剪枝。has_infrequent_subset 子程序完成非频繁子集的测试。生成所有频繁项集的 apriori 算法程序如下:

```
L1 = { 1-itemsets};
for (k=2; Lk-1≠Φ; k++) do
begin
        Ck=apriori_gen(Lk-1,min_sup);      //新的候选集
        for all transactions t∈D do
        begin
          Ct=subset(Ck,t);      //事务 t 中包含的候选集
          for all candidates c∈Ct do
             c.count++;
        end
        Lk={c∈Ck|c.count≥min_sup}
end
Answer=∪Lk;

Procedure apriori_gen( Lk-1,min_sup)
Ck=Φ
for each itemset li∈Lk-1
        for each itemset lj∈Lk-1
                if(li[1]=lj[1]) ∧ (li[2]=lj[2]) ∧ … ∧ (li[k-2]=lj[k-2]) ∧ (li[k-1]<
                lj[k-1])
            then
                begin
                c = li join lj
                if has_infrequent_subset(c, Lk-1)
                      delete c;
                else add c to Ck;
            end
return Ck;

        Procedure has_infrequent_subset(c, Lk-1)
for each (k-1)-subset s of c
        if s ∉ Lk-1 then
```

```
        return TRUE;
    return FALSE;
```

首先产生频繁 1-项集 L_1,然后是频繁 2-项集 L_2,直到有某个 r 值使得 L_r 为空,算法停止。这里在第 k 次循环中,过程先产生候选 k-项集的集合 C_k,C_k 中的每一个项集是对两个只有一个项不同的属于 L_{k-1} 的频繁集做一个连接来产生的。C_k 中的项集是用来产生频繁集的候选集,最后的频繁集 L_k 必须是 C_k 的一个子集。C_k 中的每个元素需在交易数据库中进行验证来决定其是否加入 L_k,这里的验证过程是算法性能的一个瓶颈。这个方法要求多次扫描可能很大的交易数据库,即如果频繁集最多包含 10 个项,那么就需要扫描交易数据库 10 遍,这需要很大的 I/O 负载。

Agrawal 等人引入了修剪技术来减小候选集 C_k 的大小,由此可以显著地改进生成所有频繁集算法的性能。算法中引入的修剪策略基于 Apriori 性质:一个项集是频繁集当且仅当它的所有子集都是频繁集。那么,如果 C_k 中某个候选项集有一个 $(k-1)$-子集不属于 L_{k-1},则这个项集可以被修剪掉不再被考虑,这个修剪过程可以降低计算所有的候选集的支持度的代价。J. Kleinberg 在文中还引入 hash 树(hash tree)方法来有效地计算每个项集的支持度。

8.2.4 基于 FP-树的关联规则挖掘算法

Apriori 算法有一些固有的缺陷:

- 可能会产生大量的候选集。当长度为 1 的频繁集有 10 000 个的时候,长度为 2 的候选集个数将会超过 10M。还有就是如果要生成一个很长的规则,要产生的中间元素也是巨大的。
- 必须多次重复扫描数据库,对候选集进行模式匹配,因此效率低下。

Jiawei Han 等人提出了一种基于 FP-树的关联规则挖掘算法 FP_growth,它采取“分而治之”的策略,将提供频繁项目集的数据库压缩成一棵频繁模式树(FP-树),但是仍然保留项集关联信息,然后将这种压缩后的数据库分成一组条件数据库,并分别挖掘每个数据库。理论和实验表明该算法优于 Apriori 算法。

1. 算法描述

算法 FP_growth 将发现所有的频繁项目集的过程分为以下两步:构造频繁模式树 FP-树;调用 FP_growth 挖掘出所有的频繁项目集。在 FP-树中,每个结点由 3 个域组成:项目名称 item_name、结点计数 count 和结点链(指针)。另外,为了方便树的遍历,利用频繁项集 L_1(1-项集),并增加“结点链”,通过结点链指向该项目在树中的出现,即结点链头 head,指向 FP-树中与之名称相同的第一个结点。

仍利用表 8.16 事务数据库例来说明 FP-树的构造过程和频繁模式挖掘过程。

(1) FP-树构造过程

数据库的第一次扫描与 Apriori 相同,它导出频繁项(1-项集)的集合,并得到支持度计数。设最小支持度为 2,频繁项的集合按支持度计数的递减顺序排序,结果表记作 L。这样,有:

$$L = \{B:7, A:6, C:6, D:2, E:2\}$$

FP-树构造如下：首先，创建树的根结点，用 null 标记。第二次扫描事务数据库。每个事务中的项按 L 中的次序处理(即按递减支持度计数排序)并对每个事务创建一个分支。

例如，第一个事务“T_1：A,B,E”，按 L 的次序包括 3 个项$\{B,A,E\}$，导致构造树的第一个分支<B：1，A：1，E：1>。该分支具有 3 个结点，其中 B 作根结点的子链接，A 链接到 B，E 链接到 A。从 L 表中结点链中，项 B、A、E 的指针分别指向树中 B、A、E 结点。

第二个事务“T_2：B,D”，按 L 的次序也是$\{B,D\}$仍以 B 开头，这样在 B 结点中产生一个分支，该分支与 T_1 项集存在路径共享前缀 B。这样，将结点 B 的计数增加 1，即(B:2)，并创造一个 D 的新结点(D:1)，作为(B:2)的子链接。

第三个事务“T_3：B,C”，同第二个事务一样处理，因为有相同的 B 为头，在 B 结点又产生一个分支，产生新结点，记为(C:1)，结点 B 的计数再增加 1(为 3)，即(B:3)。

第四个事务“T_4：A,B,D”，按 L 的次序为$\{B,A,D\}$。在 FP-树中 B、A 已有结点，将共享前缀路径，从 A 结点分支产生 D 的另一新结点，记为(D:1)，共享结点 B、A 的计数均增加 1，即(B:4)，(A:2)。此(D:1)结点用指针指向前面产生的(D:1)结点，在 L 表中结点链接中指针指向该(D:1)结点。

第五个事务“T_5：A,C”，按 L 表的次序为$\{A,C\}$。在 FP-树中，由于该事务不含 B 结点，不能共享 B 分支。从 null 结点产生 FP-树的第二个分支，建新 A 结点，记为(A:1)，由该结点产生分支，建新 C 结点，记为(C:1)。由于 B 分支中有(A:2)结点。这样，从(A:2)结点用指针指向此(A:1)结点，B 分支中有(C:1)结点，它用指针指向此(C:1)结点。

第六个事务“T_6：B,C”，同第三个事务那样，沿 FP-树的 B-C 分支的结点计数各增加 1，变为(B:5)和(C:2)。

第七个事务“T_7：A,C”，同第五个事务那样，沿 FP-树的 A-C 分支的结点计数各增加 1，变为(A:2)和(C:2)。

第八个事务“T_8：A,B,C,E”，按 L 表的次序为$\{B,A,C,E\}$，可沿分支 B-A 方向，在 A 结点处新建分支，建 C 结点，记为(C:1)，由该结点再建分支，建 E 结点，记为(E,1)，前面 B、A 结点计数各增加 1，变为：(B:6)、(A:3)。FP-树中原 E 结点(E:1)中的指针指向该(E,1)结点。

第九个事务“T_9：A,B,C”，按 L 表的次序为$\{B,A,C\}$，同第八个事务那样，分支 B-A-C 方向且已有结点，分别对 B、A、C 3 个结点计数增加 1，变为(B:7)、(A:4)、(C:2)。最终的 FP-树如图 8.5 所示。

从 FP-树可以看出，从 L 表的结点链的指针开始，指向 B 结点，它的计数器为 7，指向 A 结点，共有两个 A 结点，累加计数为 6；指向 C 结点，共有 3 个 C 结点，累加计数为 6；指向 D 结点，共有 2 个 D 结点，累加计数为 2；指向 E 结点，共有 2 个 E 结点，累加计数为 2。这样，频繁模式都在 FP-树中表现出来。

(2) 频繁模式挖掘过程

从 FP-树中挖掘频繁模式，先从 L 表中最后一项开始。E 在 FP-树有 2 个分支，路径为<BAE:1>和<BACE:1>。以 E 为后缀，它的两个对应前缀路径是(BA:1)和(BAC:1)，它们形成 E 的条件模式基。它的条件 FP-树只包含单个路径<B:2，A:2>，不包含 C，因为

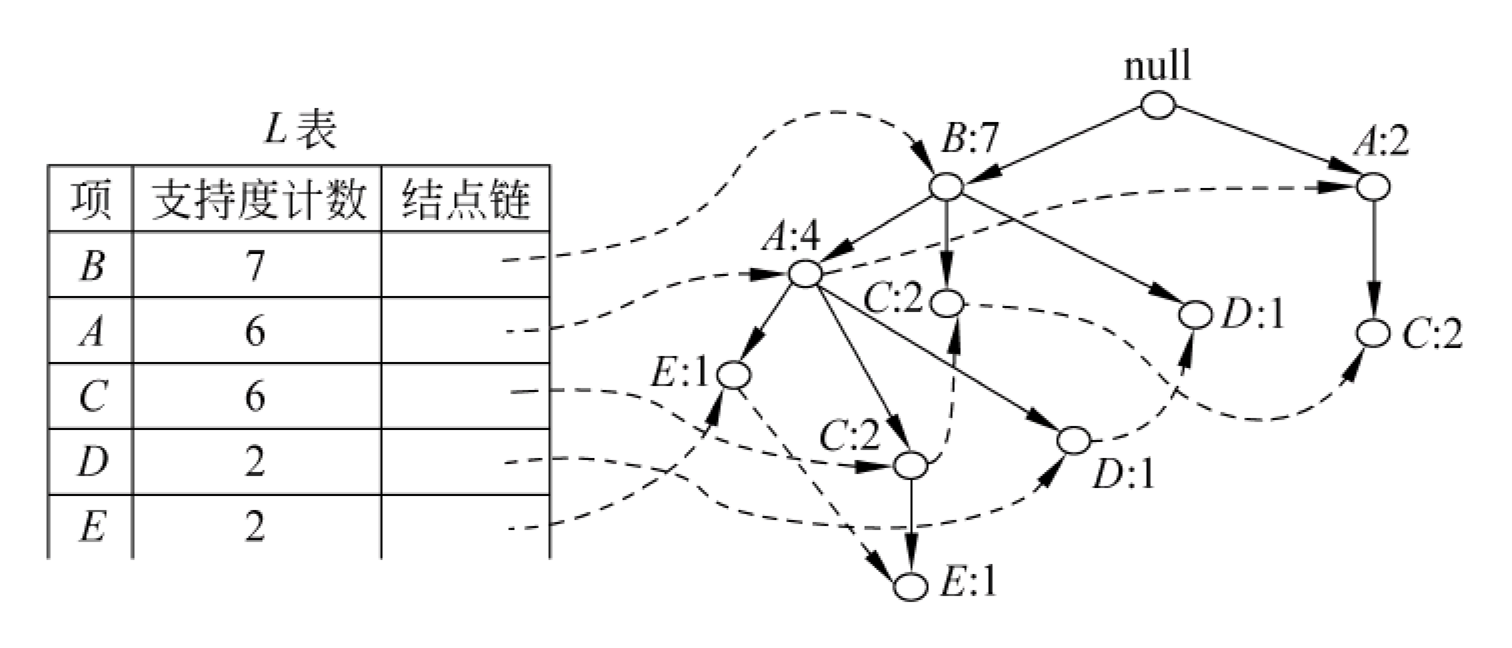


图 8.5 表 8.16 事务数据库的 FP-树

它的支持度计数为 1,小于最小支持度计数。该单个路径产生频繁模式的所有组合:{BE:2,AE:2,BAE:2}。

对于 *D*,它的两个前缀形成条件模式基{(BA:1),(*B*:1)},产生一个单结点的条件 FP-树(*B*:2),并导出一个频繁模式{BD:2}。

对于 *C*,它的条件模式基是{(BA:2),(*B*:2),(*A*:2)},它的条件 FP-树有 2 个分支(*B*:4,*A*:2)和(*A*:2)。它的频繁模式集为 {BC:4,AC:4,BAC:2}。

对于 *A*,它的条件模式基是{(*B*:4)},它的 *FP*-树只包含一个结点(*B*:4),产生一个频繁模式{BA:4},如表 8.17 所示。

表 8.17 利用 FP-树挖掘频繁模式

项	条件模式基	条件 FP-树	频繁模式
E	BA:1,BAC:1	(*B*:2,*A*:2)	BE:2,AE:2,BAE:2
D	BA:1,*B*:1	(*B*:2)	BD:2
C	BA:2,*B*:2,*A*:2	(*B*:4,*A*:2)(*A*:2)	BC:4,AC:4,BAC:2
A	*B*:4	(*B*:4)	BA:4

2. 基于 FP-树算法

(1) 构造频繁模式树算法

① 扫描事务数据库 *D* 一次。收集频繁项的集合(1-项集)以及相应的支持度。按照支持度降序排序,构成频繁项表 *L*。

② 创建 FP-树的根结点,以 null 标记。对于 *D* 中的每个事务 *T*,进行如下处理:选择 *T* 中的频繁项目,并按照 *L* 中的次序排列。设排列之后的频繁项表为[*p*|*P*],其中 *p* 是第一个项目,*P* 是剩余的项目表;如果[*p*|*P*]非空,调用 insert_tree([*p*|*P*],*T*)。

insert_tree([*p*|*P*],*T*)的执行过程如下:

如果 *T* 有子女 *N* 使得 N.item_name = p.item_name,则 *N* 的计数加 1;否则创建一个新结点 *N*,将其计数设置为 1,链接到它的父结点 *T*,并且通过结点链将其链接到具有相同 item_name 的结点。如果 *P* 非空,递归地调用 insert_tree(*P*,*T*)。

(2) 挖掘频繁项目集算法

FP-树的频繁项目集挖掘通过调用 FP_growth(FP-tree,null)实现。该实现过程如下:

Procedure FP_growth(Tree ,α)

① 如果 Tree 含单个路径 P,则

② 对于路径 P 中的每个组合(记作 β)

③ 产生模式 $\beta \cup \alpha$,其支持度 support $=$ β 中结点的最小支持度

④ 否则对于在 Tree 头部的每个 a_i

⑤ 产生一个模式 $\beta = a_i \cup \alpha$,其支持度 support $=$ a_i的支持度

⑥ 构造 β 的条件模式基,然后构造 β 的条件 Tree_β

⑦ 如果 Tree_β非空,则调用 FP_growth(Tree_β,β)

FP_growth 方法将发现长频繁模式的问题转换为递归的发现一些短模式,然后连接后缀。它使用最不频繁的项作后缀,提供了非常好的选择性,大大降低了搜索开销。

对 FP_growth 算法的性能研究表明:对于挖掘长的和短的频繁模式,它都是有效的和可伸缩的,并且大约比 Apriori 算法快一个数量级。

3. 示例说明

例如,假设有 10 个事务的数据库 D,项目集合$\{a,b,c,d,e,f,g,h,i\}$,最小支持度为 20%,如表 8.18 所示。

表 8.18 事务数据库

TID	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
项集	e	a,c,g,i	d,h	b,d	d,e	a,c,e,i	a,c,e,f,i	a,e,g	a,c,e,i	c,e,g

数据库 D 对应的频繁模式树 FP-树如图 8.6 所示。

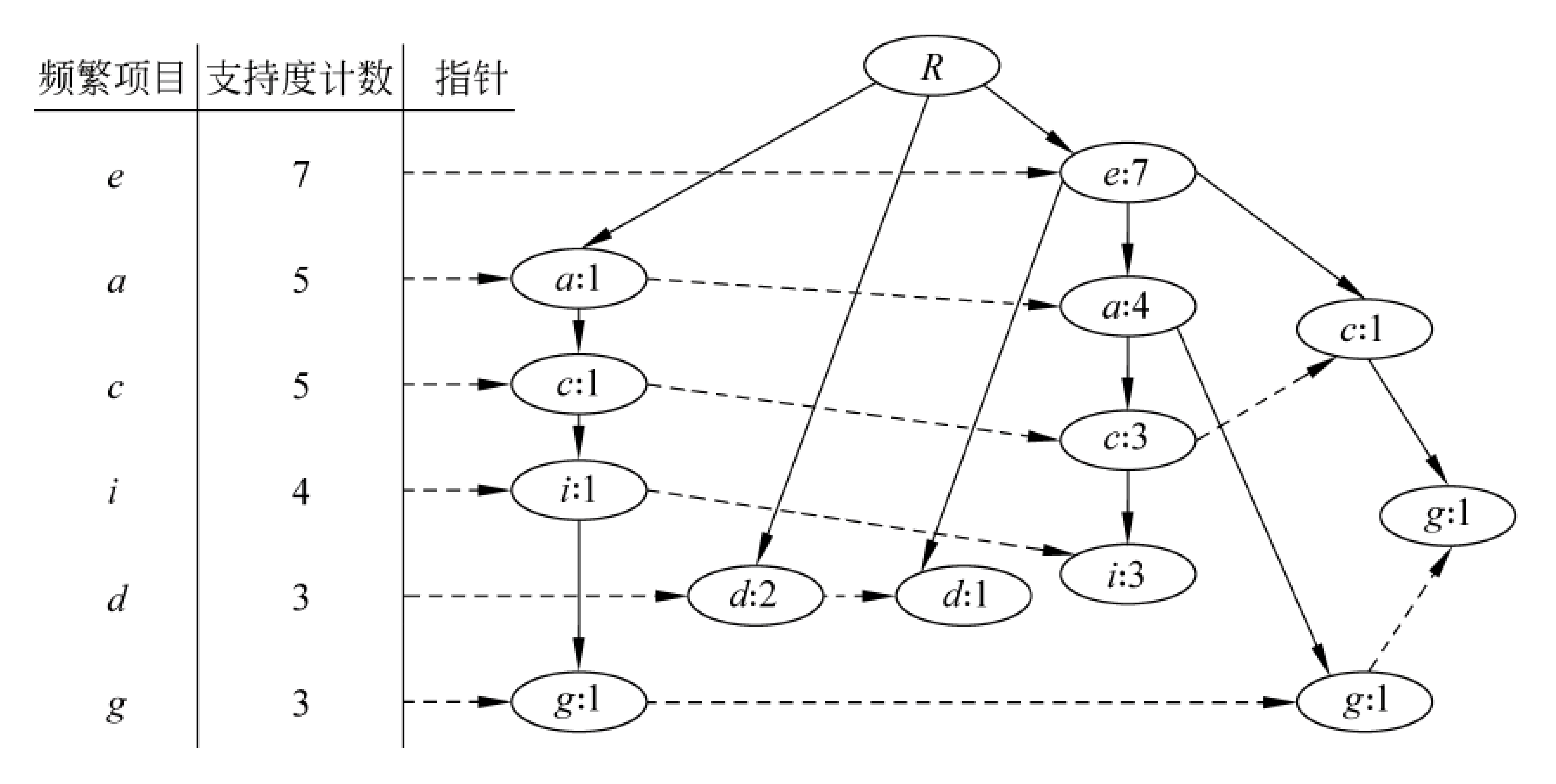


图 8.6 FP-树实例

使用 FP_growth 算法,可以得到数据库 D 的频繁项目集为$\{\{e\}:7,\{a\}:5,\{c\}:5,\{i\}:4,\{d\}:3,\{g\}:3,\{a,c\}:4,\{a,e\}:4,\{a,g\}:2,\{a,i\}:4,\{c,e\}:4,\{c,g\}:2,\{c,i\}:4,\{e,g\}:2,\{e,i\}:3,\{a,c,e\}:3,\{a,c,i\}:4,\{a,e,i\}:3,\{c,e,i\}:3,\{a,c,e,i\}:3\}$,其中 b、f、h 不是频繁项集。

习　　题

1. 说明等价关系、等价类以及划分的定义。

2. 说明集合 X 的上、下近似关系定义。

3. 说明正域、负域和边界的定义。

4. 说明粗糙集定义和确定度定义。

5. 什么是属性约简？

6. 什么是属性集的核？

7. 请用粗糙集的条件属性，相对于决策属性的约简定义，对于两类人数据库表 6.3（第 6 章）进行属性约简计算。

8. 说明条件属性 C 与决策属性 D 之间的依赖度 $\gamma(C,D)$ 的含义是什么？

9. 依赖度 $\gamma(C,D)$ 的性质是什么？

10. 属性 a 的重要度 SGF(a,C,D) 的含义是什么？

11. 最小属性集的概念是什么？

12. 在数据库中获得最小属性集的步骤是什么？

13. 如何利用集合之间的上下近似关系获得规则？

14. 规则的支持度和可信度的含义是什么？

15. 关联规则的兴趣度定义是什么？说明兴趣度的作用。

16. 数据库有如下 4 个事务。设最小支持度为 50%。使用 Apriori 算法找出所有的频繁项目集。

TID	项
T_1	A, C, D
T_2	B, C, E
T_3	A, B, C, E
T_4	B, E

17. 实现 Apriori 算法，说明 Apriori 算法的主要系统开销在哪里？

18. 对上述事务集，使用 FP_树算法找出所有的频繁项目集，并比较二者在性能上的差异。

19. 对表 8.18 事务数据库，利用 FP-树算法进行详细计算，得出图 8.6 的 FP-树。

20. 对上题得出的频繁项集求出关联规则。

第9章　公式发现

9.1　公式发现概述

9.1.1　曲线拟合与公式发现

在科学发展史上，各种物理学、化学、天文学中的自然规律都是著名科学家对大量的实验数据进行深入的研究，最后得到了自然规律，如牛顿三大定律、万有引力定律、开普勒行星运行定律等。这些自然定律是科学发展和社会进步的奠基石。

自然界存在着无数的规律，除了已被发现的外，还有很多规律需要人们去发现。在大量的工程问题中，同样存在着大量的实验数据需要人们去寻找它们的规律性。在找到完全精确的规律性之前，一般用经验性规律（带有一定的误差）来代替，去完成工程计算、设计和施工。经验规律的发现一般是由有经验的工程师来完成的。

1. 数值计算方法中的曲线拟合

随着计算机的出现，发展了数据拟合技术。它是数值计算的重要分支。数据拟合是利用科学试验中得出的大量测量数据，去求得自变量和因变量的一个近似公式。

例如，已知 N 个点 (x_i, y_i) 去求得自变量 x 和因变量 y 的一个近似表达式 $y=\phi(x)$。

曲线拟合问题的特点在于，被确定的曲线原则上并不特别要求真正通过给定的点，只要求它尽可能从给定点的附近通过。对于含有观测误差的数据来说，不过点的原则显然更为适合。因为它可以部分抵消数据中含有的观测误差。给出它们一般的近似的数学公式有

$$y^* = a_0 + a_1\phi_1(x) + a_2\phi_2(x) + \cdots + a_k\phi_k(x) \tag{9.1}$$

在曲线拟合中，$\phi_k(x)$ 一般取 x^k 或者是正交多项式。其中 $a_0, a_1, a_2, \cdots, a_k$ 各个系数的确定常用的是最小二乘法，即使各点的误差平方和最小：

$$\begin{aligned}\phi(a_0, a_1, \cdots, a_k) &= \sum_{i=1}^{n}(y_i - y_i^*)^2 \\ &= \sum_{i=1}^{n}(y_i - (a_0 + a_1\phi_1(x_i) + a_2\phi_2(x_i) + \cdots + a_k\phi_k(x_i)))^2 \\ &\to \min\end{aligned} \tag{9.2}$$

对于如何选择 $a_0, a_1, a_2, \cdots, a_k$ 使误差平方和最小，可以用数学分析中求极值的方法，即函数 $\phi(a_0, a_1, a_2, \cdots, a_k)$ 对 $a_0, a_1, a_2, \cdots, a_k$ 求偏微商，再使偏微商等于零，得到 $a_0, a_1, a_2, \cdots, a_k$ 应满足的方程：

$$
\begin{cases}
\partial\phi/\partial a_0 = -2\sum_{i=1}^{N}(y_i - a_0 - a_1\phi_1(x_i) - \cdots - a_k\phi_k(x_i)) = 0 \\
\partial\phi/\partial a_1 = -2\sum_{i=1}^{N}(y_i - a_0 - a_1\phi_1(x_i) - \cdots - a_k\phi_k(x_i)) \cdot \phi_1(x_i) = 0 \\
\cdots\cdots \\
\partial\phi/\partial a_k = -2\sum_{i=1}^{N}(y_i - a_0 - a_1\phi_1(x_i) - \cdots - a_k\phi_k(x_i)) \cdot \phi_k(x_i) = 0
\end{cases}
\tag{9.3}
$$

求得这组方程的解$\{a_i\}$，即可得拟合公式(9.1)。

用多项式作逼近公式：

$$
y = a_0 + a_1x^1 + a_2x^2 + \cdots + a_kx^k \tag{9.4}
$$

根据数学定理，k越大(x^k的次数越高)，逼近的精度越高。但实际计算表明，k过大，不但求解过程中容易发生病态等麻烦情况，而且得到的多项式尽管在各x_i处的值与y_i很接近，但其他地方却产生不合理的波动现象。

为克服这方面的困难，取更一般的情况，即用正交多项式$\Phi_k(x)$代替x^k，它本身是k次多项式，典型的如勒让德多项式。下面用一个例子来说明。

例如，在某一个化学反应里，根据实验所得分解生成物的浓度与时间的关系如表9.1所示。

表 9.1　浓度与时间的关系数据

时间 t	0	5	10	15	20	25	30	35	40	45	50
浓度 y	0	1.27	2.16	2.86	3.44	3.87	4.15	4.37	4.51	4.60	4.66

由于用简单的多项式作逼近公式，得不到理解的精度，采用勒让德多项式来作逼近公式。在此，用5次正交勒让德多项式作为y的近似公式：

$$
y = \phi_5(x) = \sum_{i=0}^{5} a_i p_{i,10}(x)
$$

其中$x=t/5$，即$x_0=0, x_1=1, \cdots, x_{10}=10$。

利用曲线拟合方式得到具体的逼近公式为

$$
\begin{aligned}
\phi_5(x) = &3.2627\times10^{-4}p_{0,10}(x) - 2.154\,55\times10^{-4}p_{1,10}(x) - 0.908\,104\times10^{-4}p_{2,10}(x) \\
&- 0.164\times10^{-4}p_{3,10}(x) - 0.0195\times10^{-4}p_{4,10}(x) - 0.0102\times10^{-4}p_{5,10}(x)
\end{aligned}
$$

其中各正交多项式为：

$$
p_{0,10}(x) = 1
$$

$$
p_{1,10}(x) = 1 - 2\cdot\frac{x}{10}
$$

$$
p_{2,10}(x) = 1 - 6\cdot\frac{x}{10} + 6\cdot\frac{x(x-1)}{10(10-1)}
$$

$$
p_{3,10}(x) = 1 - 12\cdot\frac{x}{10} + 30\cdot\frac{x(x-1)}{10(10-1)} - 20\cdot\frac{x(x-1)(x-2)}{10(10-1)(10-2)}
$$

$$
\begin{aligned}
p_{4,10}(x) = &1 - 20\cdot\frac{x}{10} + 90\cdot\frac{x(x-1)}{10(10-1)} - 140\cdot\frac{x(x-1)(x-2)}{10(10-1)(10-2)} \\
&+ 70\cdot\frac{x(x-1)(x-2)(x-3)}{10(10-1)(10-2)(10-3)}
\end{aligned}
$$

$$p_{5,10}(x)=1-30\cdot\frac{x}{10}+210\cdot\frac{x(x-1)}{10(10-1)}-560\cdot\frac{x(x-1)(x-2)}{10(10-1)(10-2)}$$
$$+630\cdot\frac{x(x-1)(x-2)(x-3)}{10(10-1)(10-2)(10-3)}$$
$$-252\cdot\frac{x(x-1)(x-2)(x-3)(x-4)}{10(10-1)(10-2)(10-3)(10-4)}$$

该逼近公式的精度是很高的，遗憾的是，此公式太复杂，计算起来繁琐，很难理解变量之间的内在关系。

曲线拟合中如何选取基函数(如勒让德多项式)的有效方法是正交筛选法。

可以说，曲线拟合方法基本上解决了在科学与工程中从大量实验数据中找出逼近公式，达到给定的精度。

数据拟合方法虽然能解决一些实际问题，但是它把寻找公式的范围限制在多项式形式之内。对正交多项式一般表示都很复杂，如勒让德多项式是由多个多项式组成。每个多项式的系数都不相同，且多项式次数逐渐增加。由正交多项式表示的逼近公式对使用者来说很不直观，建立不起各个变量之间的直观概念。

2. 发现学习

随着人工智能技术的发展，近 10 年来，机器发现技术得到发展。比较典型的系统有科学定律发现系统 BACON、数学概念发现系统 AM 等。它们都产生了巨大的影响。

对于科学发现的自然规律，用数据拟合的方法在计算机上是绝对得不出来的，只能采用新的途径，这就需要用人工智能技术来完成。BACON 系统就是在这种思想指导下产生的。

发现学习是从一组观测结果或数据利用启发式求出这些数据的一个或多个规律。

例如容器中的气体，人们能够观察到的具体数据是温度(T)、体积(V)、压强(P)和克分子个数(N)。它们之间的规律性是这些属性项之间的关系式：PV/NT＝常数。公式发现就是找出能够解释给定数据集合的最本质的规律性。

发现学习有两种方式：数据驱动方式的公式发现和模型驱动方式的概念发现。

数据驱动方式的公式发现是根据在搜索数据中所发现的数据规律性，采用不同的启发式发现动作，在一系列发现动作之后形成所发现的公式规律。BACON 系统和 FDD 系统是数据驱动的公式发现系统。

模型驱动方式的概念发现的典型例子是数学概念发现系统 AM。它包括了各种各样的搜索法(242 个启发式规则)指导在数据领域中的搜索，从集合、表、项等 1000 多个基本数学概念出发，AM 使用具体化、一般化、类比、复合等操作去产生新的数学概念，如得出自然数、质数等重要的数学概念。AM 系统还找到了与这些概念有关的定性规律，如惟一因子分解定理等。

9.1.2 启发式与数据驱动启发式

1. 启发式

启发式是人工智能的重要方法。启发式的基本定义是：能够建议合情的行动和避免不

合情的行动的知识。

通过深入的研究，对启发式有了更深入的了解。形成了对启发式的如下新观点。

(1) 通过使用启发式规则，能开发新的知识领域

通过使用这些既能建议合情的行动，又能排除不合情的行动的启发式集，可以发现一些全新的概念及其关系。

(2) 当新的知识领域产生和演变时，需要新的启发式

当引入一些新的建议、定理、技术、规范或观察到的现象后，这一领域可能随之改变，用于处理这一领域的启发式也会变化。例如，观察一个用于制定从旧金山到伦敦的旅行计划的启发式集，近些年来，加入了许多新规则，而修改了许多旧规则。

(3) 能用启发式开发新的启发式

启发式本身的生长通过启发式来引导。为了做到这一点，需要很多类型的启发式(如一般的或专用的等)、用于启发式的知识表示以及关于启发式属性的假设等。

(4) 当新的知识领域产生和进化时，需要新的知识表示。新的知识表示也能由启发式产生。

2. 数据驱动启发式

典型的 BACON 系统采用了数据驱动启发式，通过启发式搜索发现科学定律(公式)。

公式发现在于分析数据(或称观测值)得出假说(或称定律)。这些假设(定律)能够解释(或概括)这些数据。

信息用不同层次的描述表示，其中最底层的可认为是数据，而最高层的可说成是假说，中间层次则是这两个概念的混合。一个层次的描述既作为它下面一层描述的假设，又作为它上面一层描述的数据。

BACON 的启发式搜索总是注意两个数值变量之间增加和减少的单调关系。考察下面一条递减关系的启发式，可叙述为：如果在某层次的描述中，因变量 y 的值随变量 x 的相应值的减少而增加，则注意 y 和 x 之间的单调减少关系，并计算 y 关于 x 的斜率。一旦某种趋向被发现，系统就计算出有关这两变量组成直线的斜率，即：

如果发现 y 是 x 的线性函数，其斜率为 m，截距为 i，BACON 就建立一个斜率变量，定义为 $(y-i)/x$，和一个截距变量为 $y-mx$。

如果截距很接近于零值，BACON 就定义一个比率变量 y/x。

如果斜率是常数，那么系统就建立两个新的变量(m 和 i)，用来定义有关变量的线性组合。

如果该斜率是变化的(它们的关系是非线性的)，那么 BACON 就根据关系的方向和所涉及的数的符号去计算有关变量的积或商，系统把这一乘积或商也同样作为一个新变量对待，一旦定义了一个新变量，它与直接观测的因变量之间没有任何差别，都作为变量，同时去发现更新的变量关系。

9.2 科学定律重新发现系统

9.2.1 BACON 系统基本原理

1. BACON 系统的思想

BACON 系统是运用人工智能技术从试验数据中寻找其规律性比较成功的一个系统，是 Pat Langley 于 1980 年研制的。它运用数据驱动方法，即使用的规则空间与假设空间是分开的。系统的规则空间包括若干精练算子，通过精练算子修改假设。所谓精练算子就是修改假设空间的子程序，每个精练算子以特定的方式修改假设空间。整个学习程序由多个精练算子组成，程序使用探索知识对提供的训练例进行分析，决定选用哪个精练算子。这类学习方法的大致步骤为：

步骤 1　收集某些训练例。

步骤 2　对训练例进行分析，决定应该使用的精练算子。

步骤 3　使用选出的算子修改当前的假设空间。

重复执行步骤 1 到步骤 3 直到取得满意的假设为止。

BACON 系统的思想是程序反复地考察数据并使用精练算子创造新项，直到创造的这些项中有一个是常数时为止。于是一个概念就用“项＝常数”的形式表示出来，其中项是变量运算的组合而形成的表达式。

2. BACON 系统主要精练算子

BACON 系统主要精练算子如下：

(1) 发现常数

当某一属性变量取某一值至少两次的时候，触发这个算子，该算子建立这个变量等于常数的假设。

(2) 具体化

当已经建立的假设同数据相矛盾时触发这一算子，通过增加合取条件的形式把假设具体化。

(3) 斜率和截距的产生

当发现两个变量是线性相互依赖时触发这一算子，它是以建立线性关系的斜率和截距作为新变量。

(4) 乘积的产生

当发现两个变量以相反方向递增但又不线性依赖时触发该算子，产生两个变量的乘积作为新变量。

(5) 商的产生

当发现两个变量以相反方向递增但又不线性依赖时触发该算子，产生两个变量的商作

为新变量。

(6) 模 n 变量的产生

当发现两个变量 v_1 和 v_2 在模某一数 n 相等时触发这一算子，产生 $v_2(\bmod n)$ 作为新变量。

9.2.2 BACON 系统实例

1. 开普勒第三定律的发现

太阳系行星运行数据包括行星运动周期 p(绕太阳一周所需的时间)和行星与太阳的距离 d(绕太阳旋转的椭圆轨道的长半轴)，在此用参照数据，以水星数据为单位标准，见表 9.2。

表 9.2 行星运行数据

行星	p	d
水星	1	1
金星	8	4
地球	27	9

利用 BACON 精练算子发现行星运行规律过程如表 9.3 所示。

表 9.3 行星运行规律发现过程

行星	p	d	d/p	d^2/p	d^3/p^2
水星	1	1	1	1	1
金星	8	4	0.5	2	1
地球	27	9	0.33	3	1

发现过程说明如下：

(1) 变量 p 和变量 d 都是递增的，建立两变量相除的新变量 d/p(第 3 列)。

(2) 变量 d 与变量 d/p 以相反方向递增，建立两变量相乘的新变量 d^2/p(第 4 列)。

(3) 变量 d/p 与变量 d^2/p 以相反方向递增，建立两变量相乘的新变量 d^3/p^2(第5 列)。

(4) 最新变量 d^3/p^2 是常数 1，发现公式为

$$d^3/p^2 = 1$$

2. 理想气体定律的发现

理想气体有 4 个变量：体积(V)、压强(P)、温度(T)和克分子个数(N)，具体数据如表 9.4 所示。

表 9.4　理想气体数据

项目	V	P	T	N
I_1	.008 320 0	300 000	300	1
I_2	.006 240 0	400 000	300	1
I_3	.004 992 0	500 000	300	1
I_4	.008 597 3	300 000	310	1
I_5	.006 448 0	400 000	310	1
I_6	.005 158 4	500 000	310	1
I_7	.008 874 7	300 000	320	1
I_8	.006 656 0	400 000	320	1
I_9	.005 324 8	500 000	320	1
⋮	⋮	⋮	⋮	⋮
I_{25}	.026 624 0	300 000	320	3
I_{26}	.019 968 0	400 000	320	3
I_{27}	.015 974 0	500 000	320	3

为了发现它们之间的规律，先取变量 T 和 N 的相同的数据（如前 3 列中 $T=300$，$N=1$），对变量 V 和 P 进行发现，由于 V、P 两变量以相反方向递增，利用 BACON 精练算子，建立两变量相乘的新变量 PV，且 PV 等于常数 2496。对于另一组相同的数据（$T=310$，$N=1$），利用相同方法得到 PV 的新常数 2579.1999。这样得到新的理想气体数据，如表 9.5 所示。

表 9.5　合并 *PV* 变量后的理想气体数据

项目	PV	T	N
I'_1	2496	300	1
I'_2	2579.199 9	310	1
I'_3	2622.399 9	320	1
I'_4	4991.999 9	300	2
I'_5	5158.399 9	310	2
I'_6	5324.799 9	320	2
I'_7	7488	300	3
I'_8	7737.599 9	310	3
I'_9	7987.2	320	3

从表 9.4 到表 9.5，合并了变量 P 和 V 成新变量 PV，它和变量 T 和 N 仍是 3 个变量。为了有效地发现它们之间的规律，仍先固定变量 N，研究变量 PV 与 T 之间的关系。表 9.5 中每 3 行数据均为 $N=1,2,3$ 时的数据。

分析在 $N=$常数的 3 行数据中，变量 PV 与 T 是以相同方向递增，利用 BACON 精练算子建立两变量相除的新变量 PV/T，且新变量等于常数（不同 N 时，PV/T 常数不同）。这样得到的理想气体数据如表 9.6 所示。

表 9.6 最新的理想气体数据

项目	PV/T	N
I_1''	8.32	1
I_2''	16.64	2
I_3''	24.95	3

对表 9.6 中数据，它是两变量 PV/T 与 N 的数据。分析两变量 PV/T 与 N 的变化关系。两变量以相同方向递增，利用 BACON 精练算子建立两变量相除的新变量 $PV/T/N=PV/(TN)$，得到常数 8.32，按 BACON 精练算子，发现公式为

$$PV/(NT) = 8.32$$

BACON 系统在发现某些科学定律上取得很大成功，但是 BACON 系统也存在很多弱点。第一个弱点是 BACON 系统对训练例所取得的具体值特别敏感，产生这种情况的原因是因为每一个精练算子都有十分具体的触发条件，训练例的值一变，或者提供训练例的次序一变，都会影响规则的触发。例如，对某一类训练例 BACON 不能发现欧姆定律，如果变量的次序安排得不够好，BACON 发现单摆定律要多花 40% 的时间。第二个弱点是 BACON 不能处理干扰性的训练例。例如，发现常数的精练算子的触发仅仅是根据某一项在两个训练例的值相等。这种触发条件显然对干扰是高度敏感的。

9.2.3 BACON 系统的进展

BACON 系统共有 5 个版本，不同的版本其规则空间也不同。

(1) BACON.1 提出了 6 条精练算子，发现了开普勒定律。

(2) BACON.2 是 BACON.1 的扩展形式，包括两条附加的运算程序，能够发现递归序列并通过计算重复差的方法产生多项式，BACON.2 的能力有很大提高，可以解决一大类序列外推的任务。

(3) BACON.3 是 BACON.1 的另一扩展形式，使用发现常数运算程序提出的假设重新构造训练例。它用不同的描述层次来表示数据，其中最低层是直接观察的，最高层对应于数据的假说，中间层相对于下层它是假说，相对于上层它是数据，它不把假说和数据截然分开。BACON.3 由大约 86 个产生式规则组成，共分 7 组，各组产生式规则负责不同的任务，有的负责直接搜索观测数据，有的负责数据的规律性，有的计算项的值，有的把新项分解为它的组成部分。

BACON.3 发现的规律有：

- 理想气体定律：$pv/(nt)=k_1$
- Coulomb 定律：$fd^2/(q_1q_2)=k_4$
- Galileo 定律：$dp^2/(lt)^2=k_5$
- Ohm 定律：$td^2/(l_c-k_6c)=k_7$

(4) BACON.4 把观察变量的组合式认为是推理项，它使用了启发式搜索方法：程序总是注意两个数值变量之间增加和减少的单调关系，如果斜率为常数，则系统建立两个新的推理项(斜率项和截距项)作为有关变量的线性组合。如果斜率是变化的(不是线性关系)，则

BACON.4 计算有关项的乘积或比值，并把这个变量当作一个新的推理项，一旦新的项确定了，就不要区别推理项和观察变量。BACON.4 递归应用同样试探规则，使系统具有相当大的发现经验规律的能力。该系统还提出了固有性质解决符号变量的处理。

BACON.4 又发现了若干自然规律：

- Snell 折射定律：$\sin(i)/\sin(r)=n_1/n_2$
- 能量守恒定律：$m_1 v_1 = m_2 v_2$
- 万有引力定律：$F=Gm_1 m_2/d_2$
- Black 比热定律：$c_1 m_1 t_1 + c_2 m_2 t_2 = (c_1 m_1 + c_2 m_2) t_f$

（5）BACON.5 用简单的类比推理发现守恒定律，对两个物体具有完全相同的有关项，BACON.5 推测最后的定律是对称的。它把各项排序，使得属于同一物体的项首先改变，一旦该物体的这些变量中发现一个不变推理项，程序就假定必有一个类似项可用于另一物体。因此，BACON.5 只须相同地改变另一个项集合中的推理项。当做了这点之后，两个高层项取不同的值，可用其他试探规则查找它们之间的关系。这样，在物理中普遍存在的对称定律可以很容易地发现。

BACON.5 发现了能量守恒定律。

9.3 经验公式发现系统

9.3.1 FDD 系统基本原理

经验公式发现系统 FDD(formula discovery from data)是我们应用人工智能技术的机器发现技术和数值计算中的曲线拟合技术以及可视化技术结合起来自行研制的系统。它是从大量试验数据中发现的经验公式，逐步完成任意函数的任意组合（线性组合、初等运算组合、复合函数运算组合等），对自然规律和经验规律的发现。

FDD 系统有 3 个版本：FDD.1、FDD.2、FDD.3。

FDD.1 系统能够发现变量取初等函数或复合函数的组合公式。FDD.2 系统能够发现变量取导数的公式。FDD.3 系统能发现多变量取初等函数或复合函数的组合公式。

1. 问题描述

给定一组可观察变量 $X(x_1, x_2, \cdots, x_n)$ 以及这组变量的试验数据 $D_i(d_{i1}, d_{i2}, \cdots, d_{in})$，$i=1,2,\cdots,m$，公式发现系统找出该组变量满足的数学关系式：$f(x_1, x_2, \cdots, x_n)=c$，其中 c 为常数，即：对于任意一组试验数据$(d_{i1}, d_{i2}, \cdots, d_{in})$均满足关系式 $f(d_{i1}, d_{i2}, \cdots, d_{in})=c$。

所找出的关系式 $f(x)$是任何形式的数学公式，包括分段函数。

对于关系式 $f(x_1, x_2, \cdots, x_n)=c$ 的复杂程度可分为：

（1）变量的初等运算：$f(x,y)=x\theta y$，其中 θ 为＋、－、*、/之一。

（2）变量的初等函数运算：$f(x)=c$，其中 $f(x)$为初等函数。

（3）初等函数的任意组合：$f(x,y)=a_1 f(x)\theta a_2 f(y)$。

(4) 复合函数的运算：$g(f(x))=c$，其中 $g(x)$、$f(x)$均为初等函数。

(5) 复合函数的任意组合：$h(a_1g_1(f(x))\theta a_2g_2(f(y)))$，其中 $h(x)$、$g(x)$、$f(x)$均为初等函数。

(6) 多个初等函数的组合：$f(x,y)=a_1f_1(x)\theta a_2f_2(x),\cdots,\theta a_kf_k(y)$，其中 $f(x)$、$f(y)$均为初等函数。

(7) 分段函数：对于不连续的点，分别用不同的函数加以描述。

以上是对两个变量的讨论。在现实世界中存在着多变量的更为复杂的关系，在公式发现过程中采用先寻找两变量的关系，再逐步扩充为多变量的关系的方法。

2. FDD.1 的设计思想

FDD.1 系统的基本思想是利用人工智能启发式搜索函数原型，寻找具有最佳线性逼近关系的函数原型，并结合曲线拟合技术及可视化技术来寻找数据间的规律性。

启发式方法是求解人工智能问题的一个重要方法。一般启发式是建立启发式函数，用以引导搜索方向，以便用尽量少的搜索次数，从开始状态达到最终状态。

FDD.1 系统在执行搜索的过程中，对原型函数的搜索以及对它们的组合函数的搜索，也是一种组合爆炸现象。为解决这一问题，在设计系统时采用了启发式方法来实现。

对某一变量取初等函数和另一变量的初等函数或该变量进行线性组合，即从原型库中选取逼近效果最好的少数几个初等函数作为基函数，并进一步形成组合函数，直至找到最后的目标函数。FDD.1 系统的启发式函数形式为

$$f(x_2)=a+bf_1(x_1) \tag{9.5}$$

线性逼近误差公式为

$$\mathrm{dt}=(a+bf_1(x_1)-f(x_2))/f(x_2) \tag{9.6}$$

总是选取 dt 最小的 $f(x_i)$ 作为继续搜索的当前结点。这一启发式函数在以后的多次应用中证明是有效的。

3. FDD.1 系统中的知识

在 FDD.1 系统中，知识采用的是产生式规则的表示形式(if…then)。主要的基本规则有如下几个。

规则 1　发现常数

当某一变量 x 取一个常数，则建立该变量等于常数的公式，即 $x=c$。

规则 2　两变量的初等运算组合

当两变量进行初等运算若等于常数，则建立该变量的初等运算关系式：

$a_1x_1\theta a_2x_2=c$，其中 θ 为＋、－、*、/之一。

规则 3　变量取初等函数

当某变量取初等函数等于常数，则建立该变量的初等函数关系式：

$f(x)=c$，其中 $f(x)$为初等函数。

规则 4　两变量取初等函数的线性组合

两变量分别取初等函数后的线性组合等于常数，则建立两变量取初等函数的线性组合

关系式：

$$a_1 f_1(x_1) + a_2 f_2(x_2) = c$$

其中 $f_1(x_1)$、$f_2(x_2)$为初等函数。

规则 5　某变量取某一初等函数与另一变量的线性组合

对某一变量 x_i 取初等函数后与另一变量 x_j 进行线性组合，若为常数，则建立关系式：

$$c_1 f(x_i) + c_2 x_j = c$$

规则 6　对某一变量 x_j取初等函数，另一变量 x_i取两个初等函数进行线性组合，若为常数，则建立关系式：

$$c_1 f_1(x_i) + c_2 f(x_i) + c_3 g(x_j) = c$$

规则 7　建立新变量(启发式 1)

若两变量的某初等运算结果接近常数，则建立新变量为该两变量的某种初等运算。

规则 8　建立某变量的某种初等函数为新变量(启发式 2)

若某变量的某种初等函数与另一变量或它的初等函数进行线性组合接近常数，则建立该变量的初等函数为新变量。

以上规则的嵌套或递归使用，将形成变量的任意函数间的任意组合。在应用规则时，利用可视化技术将减少各种函数和各种运算的选取，大大节省了搜索时间。

9.3.2　FDD.1 系统结构

1. 系统结构图

FDD.1 系统结构图如图 9.1 所示，该系统由试验数据输入、数据生成器、公式发现控制、可视化过程、数据项、原型选择、公式生成、误差分析、循环控制、公式输出与可视化显示 10 个模块以及原型算法库、数据库、知识库、公式库 4 个库组成。

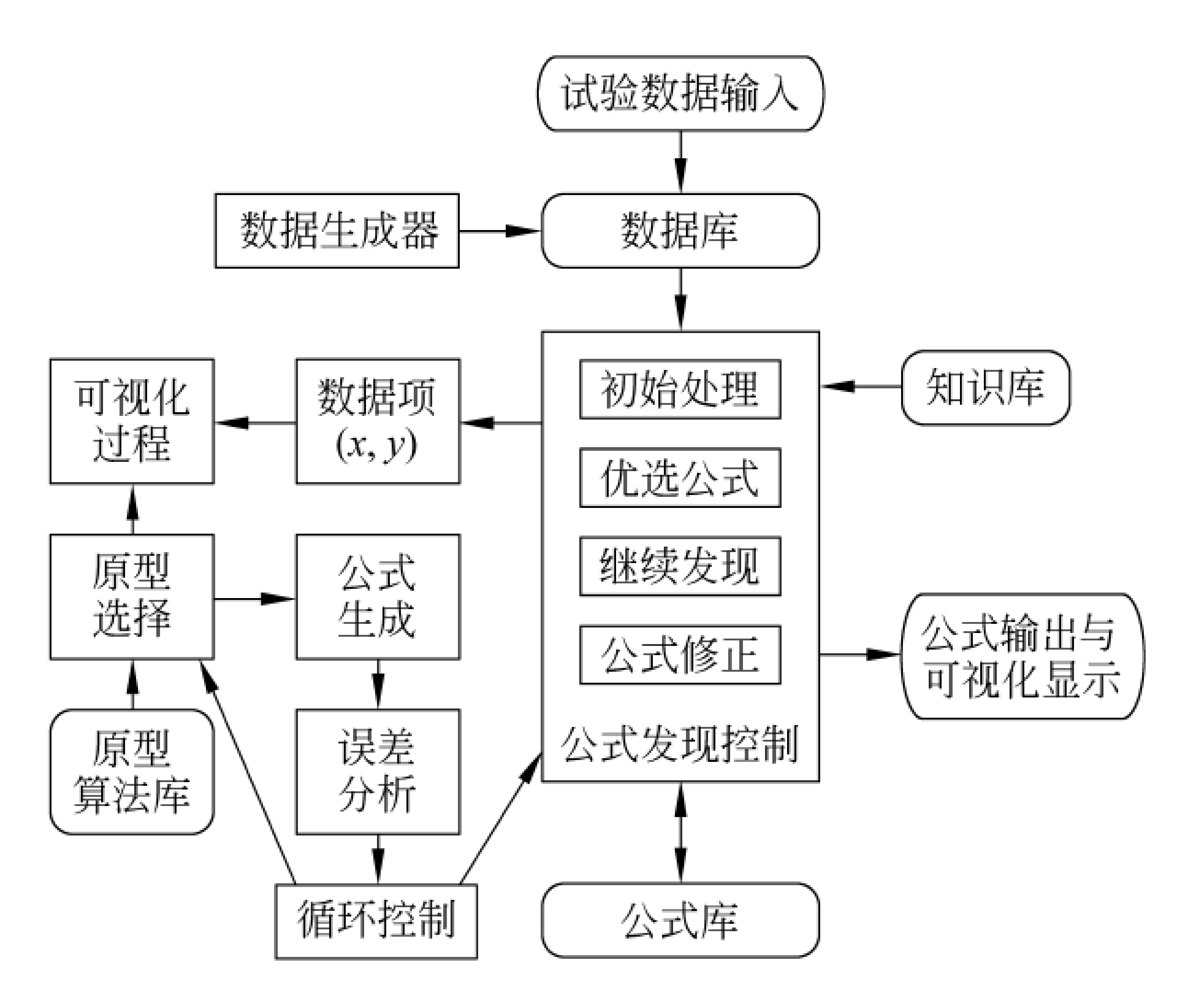


图 9.1　FDD.1 系统结构图

2. 各模块说明

(1) 试验数据输入(input data)

提示用户输入试验数据。

(2) 数据生成器(generator)

此模块用于测试系统效果。给定一个已知公式后,能生成一批数据,FDD.1 系统的核心程序将利用这些数据找出给定的公式,从而达到测试系统的公式发现能力的效果。此模块是一个可独立执行模块。

(3) 数据库(database)

数据库存放待处理的变量数据,一般是科学和工程实验数据。公式的正确与否与数据的规律性和充分性密切相关。系统本身可提供直接输入数据的功能,用户可在系统的提示下将数据输入。也可用数据生成器为系统提供数据,系统将其按一定的格式存储起来,存放在数据库中。数据库中有一个缓冲区,供系统运行时存放中间变量数据以及实现数据的移动和变化。

(4) 可视化过程

此模块又分成 3 个子模块:

- 描绘试验数据的变化趋势。
- 描绘出原型算法库中各函数原型的变化规律。此子模块具有很大的灵活性,用户可根据需要随意调用所选择原型,以描绘其变化趋势。
- 描绘所发现的公式的变化规律与原始数据之间的误差分布状况。

(5) 公式发现控制模块

此模块是 FDD.1 的核心部分,它主要是利用知识库中的知识:优选函数原型、控制继续发现、公式修正等。它包含:初始处理、优选公式、继续发现、公式修正 4 个子模块。下面对这 4 个子模块的功能进行说明:

① 初始处理。此模块的主要功能有两个,其一是根据具体情况对用户所提供的数据进行初步处理;其二是在多变量中选择两个变量以及向多变量的过渡处理。

② 优选公式。其主要功能是对公式库中提供的公式根据其误差逼近情况来优选函数原型,对函数原型一般选择 2～3 个。

③ 继续发现。此模块将根据误差分析情况完成如下功能:

- 建立新变量。
- 颠倒变量关系。
- 对所选择的函数原型进行组合。

④ 公式修正。这是在输出公式之前所必经的一个过程,此过程将根据用户提供的误差要求决定是否对系统所发现的公式进行修正。若不必修正则将公式送入“公式输出”与“可视化”模块。否则对公式进行修正。目前系统提供了 3 种公式修正方法。如下所述:

- 调和级数回归。由数学分析可知,对任意周期函数 $y=f(x)$,可以用三角函数的傅里叶级数来逼近,即

$$y = \phi(x) = a_0 + \sum_{j=1}^{m}(a_j\cos(jx) + b_j\sin(jx)) \tag{9.7}$$

将 n 组试验数据(x_i, y_i)代入上式,各点误差值以调和函数方程式的形式表示为

$$y_i = a_0 + \sum_{j=1}^{m}(a_j\cos(jx_i) + b_j\sin(jx_i)) \tag{9.8}$$

$$i = 1,2,\cdots,m; \quad j = 1,2,\cdots,m$$

可以按最小二乘原理求出调和级数中各未知系数。

- 用直线来描述误差:此算法和公式生成模块的直线拟合法类似。
- 神经元网络方法逼近误差函数:利用神经元网络中函数式网络对误差函数进行计算,求出网络权值,使函数型网络逼近该误差函数。函数型网络选取的函数为

$$\sin(2\mathrm{k}\pi x)、\cos(2\mathrm{k}\pi x) \quad k = 1,2,\cdots,n$$

(6) 数据项

程序中的两个指针变量用以存放在多个变量中所选择出的两个变量的实验数据。

(7) 原型选择

此过程通过调用原型算法库、可视化过程及误差分析模块提供的误差进行函数原型的选择。有两种选择方式:

- 由用户指定选择。
- 通过循环控制进行顺序选择。

(8) 公式生成

此模块主要应用数值分析中的曲线拟合技术求出拟合公式的系数,同时生成公式。

(9) 误差分析模块

此模块的主要功能是对公式生成模块提供的公式,计算相对误差并对各公式误差进行比较。

(10) 循环控制模块

此模块设有一个控制开关,对“原型选择”和“公式发现控制”两个过程进行循环运行。

(11) 公式输出与可视化显示

此过程是系统所要执行的最后一步,当公式发现控制模块决定最终输出公式后执行此模块,输出公式并进行可视化显示。这样用户可以很直观地阅读公式,并了解所发现的公式逼近实验数据的情况。

(12) 原型算法库

原型是构成数学公式的基本单元,原型算法库所包括的原型决定了系统的发现能力。本系统的函数原型由基本原型和组合原型构成。

基本原型由初等函数组成,如:x、x^2、x^3、x^{-1}、x^{-2}、$\mathrm{sqrt}(x)$、$x^{1/3}$、$\log(x)$、$\exp(x)$、$\sin(x)$、$\cos(x)$等。

组合原型由初等函数的初等运算组合而成,如:$x\sin(x)$、$x\cos(x)$、$x\exp(x)$、$x\lg(x)$、$x^{-1}\lg(x)$、$x^{-1}\exp(x)$、$1/\lg(x)$、$1/\mathrm{sqrt}(x)$、$\sin(x)+\cos(x)$等。

在原型算法库中,每个原型都给出了一个算法,只不过每个算法的程序结构都非常相似。

用户还可以根据需要随意增加、删除原型，在程序运行过程中给出了一个控制参数，用户可通过它来调用所需算法。

(13) 知识库

知识库中的知识用于构造和发现关系式。

(14) 公式库

公式库用来存放在系统搜索过程中初步选择的原型函数组成的公式，以备公式发现控制模块使用。

公式库中的公式包含两个变量取某原型函数的线性组合以及该公式的逼近误差。在搜索过程中，每当发现一个比较可行的公式或函数原型，便将其送入公式库等待下一步的选择，每一轮选择之后便把落选的公式剔除出公式库，直至发现满意的公式为止。

9.3.3 FDD.1 系统实例

1. 行星运动开普勒第三定律的重新发现

(1) 原始数据

原始数据如表 9.7 所示。

表 9.7 行星运行的近似数据

距离 d	1	4	9	16	25	36	49	64	81	100
周期 p	1	8	27	64	125	216	343	512	729	1000

(2) 开普勒第三定律搜索树

对于行星绕太阳运动的开普勒第三定律，BACON 系统利用变量的乘除运算，使得到的新变量趋向常数的思想，对该定律重新发现。我们利用变量取初等函数的线性组合趋向直线方程的思想，对该定律也重新发现，公式发现的搜索树如图 9.2 所示。从搜索过程可见，FDD.1 系统的公式的发现过程与 BACON 系统的公式发现过程是完全不同的。

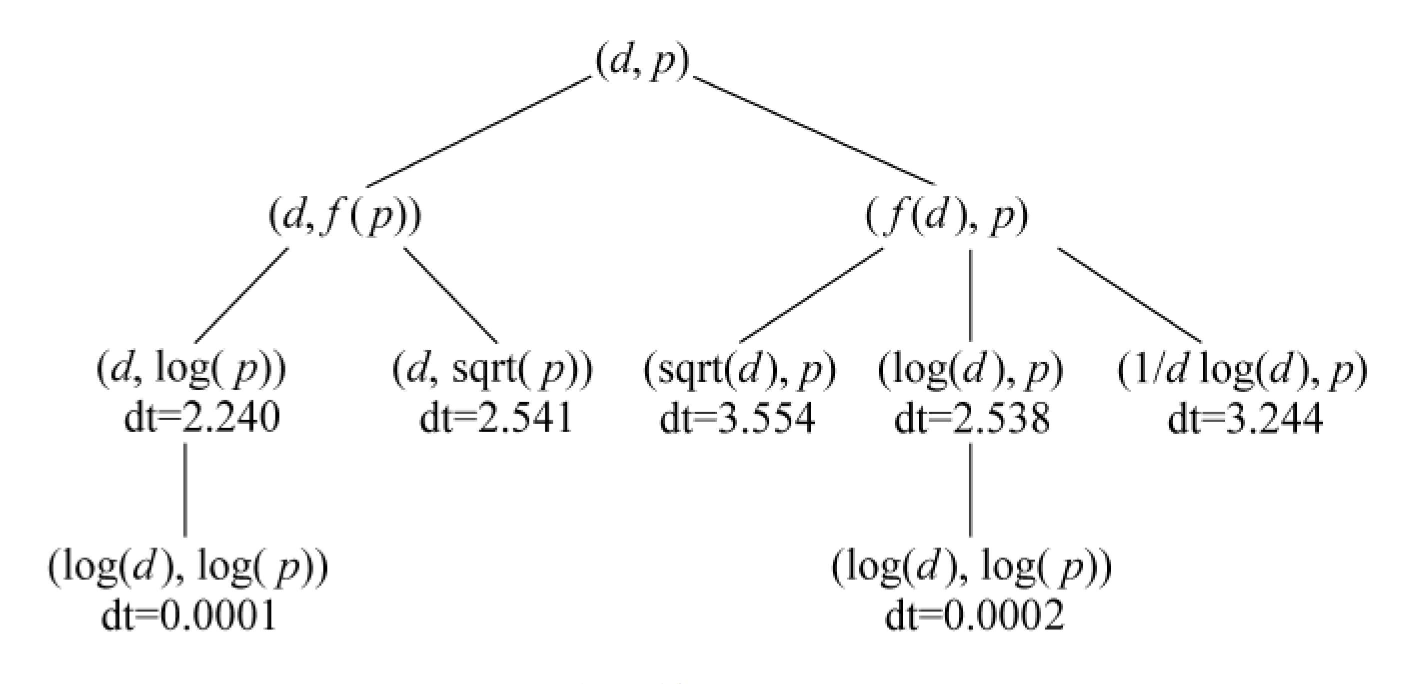


图 9.2 开普勒第三定律公式发现图

公式发现搜索树中有两个分支，左分支路径为：先固定 d，对变量 p 求各原型函数 $f(p)$，用 d 和 $f(p)$ 拟合线性方程 $f(p)=a+bd$，其中 a、b 是常数，求逼近 $f(p)$ 的相对误差，选误差最小的函数为 $\log(p)$，误差为 2.240，建立新变量 $p'=\log(p)$，并固定它，再对 d 变量

求各原型函数 $g(d)$，对 $\log(p)$和 $g(d)$拟合线性方程，并求逼近 $g(d)$的相对误差，选取误差最小者为 $\log(d)$，误差为 0.00001，调用公式生成模块，求得公式及系数，公式为

$$\log_{10}(d) = 0.0 + 0.666\,666\,667\log_{10}(p) \tag{9.9}$$

即

$$d^3 = p^2$$

从右分支树也可发现开普勒第三定律，这里不再详述。

2. 实例数据的公式发现

例如，炼钢厂出钢时所用盛钢水的钢包，在使用过程中由于钢液及炉渣对包衬耐火材料的侵蚀，使其容积不断增大，钢包的容积与相应的使用次数(即包龄)的数据如表 9.8 所示。

表 9.8 钢包容积数据

使用次数 x	容积 y	使用次数 x	容积 y
2	106.42	11	110.59
3	108.20	14	110.60
4	109.58	15	110.90
5	109.50	16	110.76
7	110.00	18	111.00
8	109.93	19	111.20
10	110.49		

对这组试验数据的搜索过程与行星运动开普勒第三定律的例子相同，这里不再详细叙述其具体发现过程，只给出了它的公式发现搜索树和最终公式形式，并与有关《计算方法引论》书中方法及结果作比较，公式发现搜索树如图 9.3 所示。

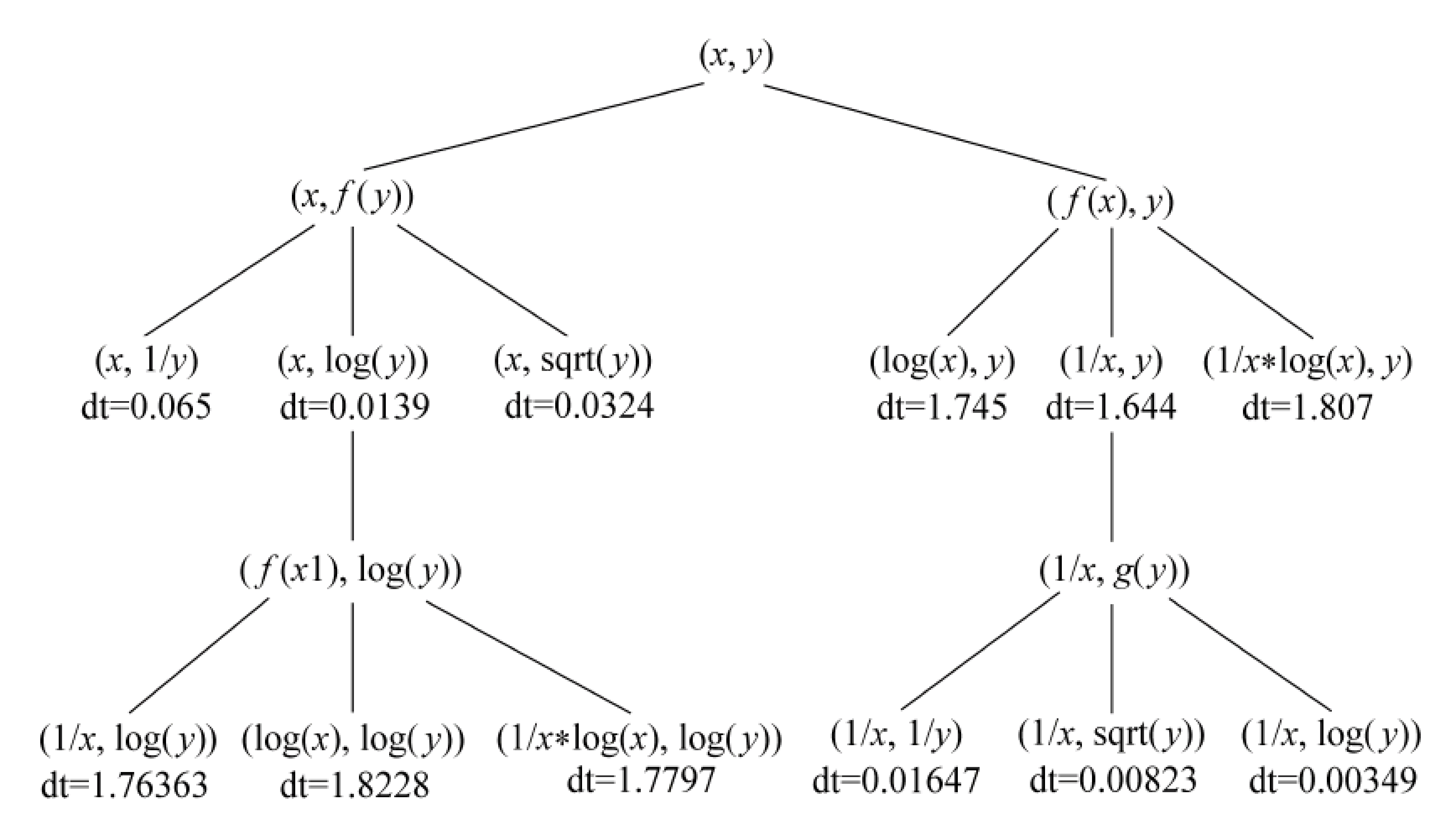


图 9.3 钢包容积变化公式发现图

从右分支开始搜索，得到了组成公式的两组基函数：$(1/x, \log(y))$ 及$(1/x, \mathrm{sqrt}(x))$，调用公式发现模块求得公式及系数，最终得到经验关系式为

$$\mathrm{sqrt}(y) = 10.559\,190\,8 - 0.471\,126\,8/x, \quad dt = 0.008\,233 \tag{9.10}$$

$$\log(y) = 2.047\,297\,5 - 0.039\,212\,4/x, \quad dt = 0.003\,49 \tag{9.11}$$

经效果分析均满足误差要求。

这样用 FDD.1 系统发现了上述两个公式。

《计算方法引论》书中所讲述的公式为

$$y = x/(0.008\,966 + 0.000\,830\,12x) \tag{9.12}$$

公式(9.12)是人们根据自己的专业知识和经验,并根据其离散点在图上分布形状选择适当的曲线公式来拟合数据,并经过一定的公式变形而得到的。从许多试验数据的分布状况,人们往往看不出它的具体规律,因此这种做法不具有普遍性,而且具有一定的盲目性。而用 FDD.1 发现经验公式并不一定要求用户的经验和专业知识,用户只提供充分的试验数据,并作一些简单的交互,FDD.1 系统很快便能发现效果良好的经验公式,这是 FDD 系统的一个显著优点。下面比较用 FDD.1 系统发现的公式和书中公式所拟合的每个点的好坏。

由公式(9.10)、式(9.11)、式(9.12)所拟合的每个点的 y 值分别用 y_1'、y_2'、y_3'表示,它们各点的值如表 9.9 所示。

表 9.9　3 个公式效果比较表

x	y_1'	y_2'	y_3'	y
2	106.58	106.58	106.60	106.42
3	108.20	108.20	109.19	108.20
4	109.02	109.02	109.01	109.58
5	109.51	109.59	109.50	109.50
7	110.08	110.08	110.08	110.00
8	110.25	110.25	110.26	109.93
10	110.50	110.50	110.51	110.49
11	110.59	110.59	110.60	110.59
14	110.79	110.79	110.80	110.60
15	110.84	110.83	110.85	110.90
16	110.88	110.87	110.89	110.76
18	110.95	110.94	110.96	111.00
19	110.98	110.98	110.99	111.20

由表 9.9 的数据可以看出,由公式(9.10)、式(9.11) 所得到的拟合值比由公式(9.12)的逼近值更加精确,这说明了 FDD.1 系统发现试验数据的经验公式是成功的。

9.3.4　FDD.2 系统

1. FDD.2 问题描述

给定两组可观察变量 $X(x_1,\ x_2)$以及这组变量的实验数据 $D_i(d_{i1}, d_{i2})$, $i=1,2,\cdots,n$,公式发现系统找出该组变量满足的数学关系式:$f(x_1,\ x_2)-c\ =\ \min$,其中 c 为常数,即:对于任意一组实验数据(d_{i1}, d_{i2}),均满足关系式$f(d_{i1}, d_{i2})-\ c=\ \min$,所找出的关系式$f(x)$是任何形式的数学公式。

对于关系式 $f(x_1,\ x_2)-c\ =\ \min$ 中的函数 f 的复杂程度可分为:

- 变量的初等运算 $f(x,y)=x\theta y$,其中 θ 是+、-、*、/之一;

- 变量的初等函数运算 $f(x)=c$，其中 $f(x)$为初等函数；
- 初等函数的任意组合 $f(x,y)=a_1 f(x)\theta a_2 f(y)$；
- 复合函数的运算 $g(f(x))=c$，其中 $g(x)$、$f(x)$均为初等函数；
- 导数处理函数。

设给出的测量数据如下：

I	1	2	…	N
X	x_1	x_2	…	x_n
Y	y_1	y_2	…	y_n

则：一阶差分：$\Delta x_k=x_{k+1}-x_k$，$\Delta y_k=y_{k+1}-y_k$，$(k=1,2,\cdots,n-1)$

二阶差分：$\Delta^2 y_k=\Delta y_{k+1}-\Delta y_k$，$\Delta^2 x_k=\Delta x_{k+1}-\Delta x_k$，$(k=1,2,\cdots,n-2)$

……

m 阶差分：$\Delta^m y_k=\Delta^{m-1} y_{k+1}-\Delta^{m-1} y_k$，在这里差分指向前差分。

一阶差商：$\delta y_k=(y_{k+1}-y_k)/(x_{k+1}-x_k)$，$(k=1,2,\cdots,n-1)$

二阶差商：$\delta^2 y_k=(\delta y_{k+1}-\delta y_k)/(x_{k+2}-x_k)$，$(k=1,2,\cdots,n-2)$

……

m 阶差商：$\delta^m y_k=(\delta^{m-1} y_{k+1}-\delta^{m-1} y_k)/(x_{k+m}-x_k)$

可以用导数表达差商，若 $f(x)$在$[a,b]$上 n 次可微，$x_1,x_2,\cdots,x_n$是$[a,b]$内的(n)个不同的点，则有 $\xi(a<\xi<b)$使 $\delta^{n-1} y=f^{(n-1)}(\xi)/(n-1)!$。

2. FDD.2 规则描述

在 FDD.2 系统中，知识同样采用的是产生式规则的表示形式(if…then)。包括 FDD.1 系统的规则外，还包括如下规则。

规则 1 差分发现常数

当某一变量差分 Δy 取一个常数 c，则建立该差分变量等于常数的公式，即 $y=a+cx$。

规则 2 差商发现常数

当两个变量差商取一个常数 c，则建立该变量导数等于常数的公式，即 $y'=c$。

规则 3 特殊函数形式导数函数

(1) 阶差(向前差分)法判定类型

- 若 $\Delta^2 y_i$=定值，则方程为 $y=a+bx+cx^2$；
- 若 $\Delta^3 y_i$=定值，则方程为 $y=a+bx+cx^2+dx^3$；
- 若 $\Delta(y_i)^{-1}$=定值，则方程为 $y^{-1}=a+bx$；
- 若 $\Delta^2(y_i^2)$=定值，则方程为 $y^2=a+bx+cx^2$；
- 若 $\Delta^2(x_i/y_i)$=定值，则方程为 $y=x/(a+bx+cx^2)$；
- 若 Δy_i成等比数列，则方程为 $y=ab^x+c$；
- 若 $\Delta\log(y_i)$成等比数列，则方程为 $\log(y)=a+bx+cx^2$；
- 若 $\Delta^2 y_i$成等比数列，则方程为 $y=ab^x+cx+d$。

(2) 差商判定类型

- 若 $\Delta\log(y_i)/\Delta\log(x_i)$=定值，则方程为 $\log y=ax^b$；
- 若 $\Delta\log(y_i)/\Delta x_i$=定值，则方程为 $y=ab^x$；
- 若 $\Delta(x_i y_i)/\Delta x_i$=定值，则方程为 $y=a+b/x$；
- 若 $\Delta(x_i/y_i)/\Delta x_i$=定值，则方程为 $y=x/(ax+b)$；
- 若 $\Delta y_i/\Delta(x_i)^2$=定值，则方程为 $y=a+bx^2$。

规则 4 两变量的导数运算组合

当某变量差分(或差商)后与另一变量进行初等运算，若等于常数，则建立该变量差分(或差商)的初等运算关系式

$$\Delta f(x_1)\theta f(x_2) = c$$

其中 θ 是+、−、*、/之一，其中 Δf 为差分或差商计算。

规则 5 两变量取导数运算的线性组合

两变量分别取导数运算后的线性组合等于常数 c，则建立两变量取导数运算的线性组合关系式

$$a_1\Delta f_1(x_1) + a_2\Delta f_2(x_2) = c$$

其中 $\Delta f_1(x_1)$、$\Delta f_2(x_2)$为导数运算。

以上规则和 FDD.1 中规则的嵌套或递归使用，将形成变量的任意函数和导数运算组合。

3. FDD.2 公式发现实例

(1) 导数函数公式的发现

x、y 为样本数据，Y 为发现的公式计算值，见表 9.10 所示。

表 9.10 导数函数公式的发现

x	1.01	2.07	2.98	7.89	7.02	6.03	6.98	8.01	9.04	9.99	11.02	12.01	12.97
y	4.61	10.51	14.65	14.61	11.08	10.2	12.6	18.27	27.3	24.46	22.08	19.72	20.93
Y	4.667	10.662	14.248	14.524	11.741	10.383	12.679	18.263	27.174	24.257	22.045	19.965	21.115

发现导数函数公式为 $y' = 1.52 - 4.34\sin(x)$，误差为 0.048。

(2) 复合函数公式的发现

数据如表 9.11 所示。

表 9.11 复合函数公式的发现

x	0.10	0.12	0.23	0.25	0.30	0.26	0.55	0.76	0.81	0.89	0.91	1.01	1.44	1.50
y	7.146	7.288	6.156	6.329	6.782	6.417	9.532	12.588	17.443	14.936	17.337	17.53	37.81	47.02
Y_1	4.899	5.044	5.924	6.10	6.561	6.190	9.371	12.51	13.385	14.921	15.334	17.59	36.50	43.98
Y_2	6.65	6.66	6.67	6.80	6.92	6.82	8.38	11.236	12.204	14.021	14.530	17.43	37.91	41.98
Y_3	7.185	7.310	6.07	6.228	6.636	6.306	9.268	12.525	17.491	17.223	17.696	18.33	37.0	40.99

发现公式为 $Y_1 = 7.94x - 11.64\log(|\cos(x)|)+4.25$，公式的误差为 0.095，如图 9.4 所示。

另外还发现两个公式：

$Y_2 = 6.639\,005\,246 + 10.471\,877\,51x^3$

$\text{sqrt}(Y_3) = 0.926\,907\,91 + 1.221\,648\,810e^x$

9.3.5 FDD.3 系统

1. 多维函数空间定义

多维函数空间由初等函数、初等函数组合、复合函数、复合函数组合、函数导数等组成。初等函数组合是初等函数之间运算组合；导数处理包括一阶差分、二阶差分、一阶差商、二阶差商等。多维函数空间的构造如下：

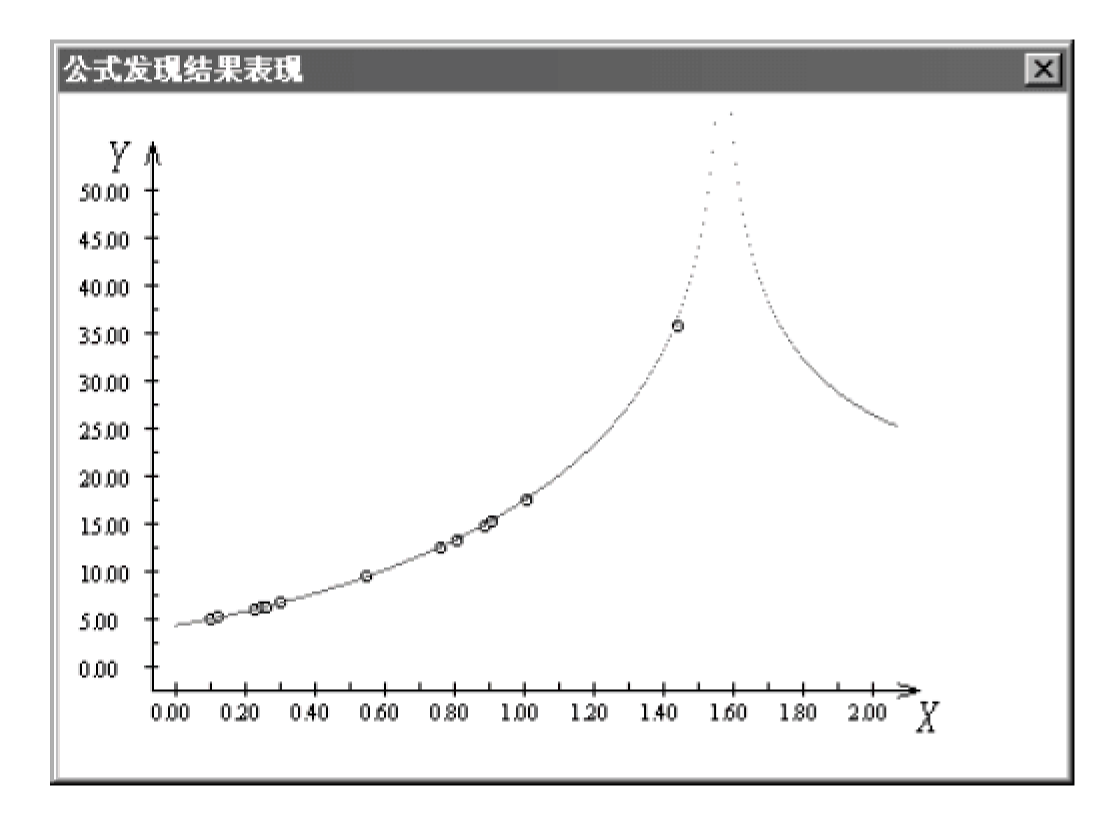


图 9.4　复合函数公式发现

定义　设多维函数空间 Ω：Ω=<P,V,C>，其中，

- $P=\{f_1, f_2, \cdots, f_m\}$是一个多元函数集，$f_i$是多元函数；
- $V=\{v_1, v_2, \cdots, v_k\}$是一个有穷变元集；
- $C=\{c_1, c_2, \cdots, c_k\}$是一个有穷常数集。

P 函数集可以包括：

- 算术运算(如＋、－、×、/等)；
- 初等函数(如 1、x^1、x^2、$x^{1/3}$、sin、cos、exp、log 等一元函数)；
- 导数函数。

2. 多维函数空间性质

从以上定义可以看出，多维函数空间具有如下性质：

性质 1　在多维函数空间中，设 $E = V \cup C$，满足条件：

(1) 对 $\forall e$，若 $e \in E$，则 $e \in \Omega$；

(2) 对 $\forall f, e_i$，若 $f \in P$，$e_i \in E$，则 $f(e_1, e_2, \cdots, e_n) \in \Omega$，$i=1,2,\cdots,n$，即函数作用于变元或常数仍然属于函数空间；

(3) 若 $p_1, p_2, \cdots, p_n \in \Omega$，则对 $\forall f \in P$，$f(p_1, p_2, \cdots, p_n) \in \Omega$，即函数作用于函数仍然属于函数空间。

性质 2　由于函数作用于变元或常数和函数作用于函数仍然是函数，故函数空间是封闭的。

对于在函数空间上的任意函数组合仍然在函数空间中，这样为计算机对函数空间的处理提供了可以递归的前提。在函数空间中的函数集合可以组成解决问题的原型库。原型库一般包括初等函数、组合函数、复合函数，还包括差分计算、差商计算以及导数计算等。

3．FDD.3 规则内容

系统中的知识采用产生式规则表示形式(if…then)，规则内容包括函数规则和控制规则。函数规则组成知识库，知识库不仅包括 FDD.1 系统规则、FDD.2 系统规则，还包括以

下规则。

(1) 函数规则(funrule)

对某一变量 x 取函数空间中的一个函数 f_i 后,与另一变量 y 的函数 f_j 进行线性组合,得到函数公式后,代入 x 和 y 的值,取函数公式两边值的误差最小,则有函数公式:

$$C_1 f_i(x) + C_2 f_j(y) = C_3, \quad f_i, f_j \in P, \quad C_1, C_2, C_3 \in C$$

(2) 函数嵌套规则

对函数规则嵌套或递归使用,将形成变量的任意组合。

(3) 误差规则(errrule)

- 误差最小规则:选择误差最小的公式进入下一次迭代。
- 误差收敛规则:保留误差减小的搜索方向,上一次迭代的误差大于目前的误差,则对于这一搜索方向予以保留。

(4) 终止规则(endrule)

终止准则由两部分组成,一是强制终止,另一个是自然终止。强制终止通过对算法参数的设定,主要是通过对迭代次数的设定完成终止准则;自然终止有两种情况组成,一种情况是找到一组满足给定误差的公式,另一种情况是判断出误差增大时,则停止该路径的搜索。

(5) 多维函数扩展规则(multirule)

① 扩展到三维函数公式的启发式规则

设给定 n 组不同的数据 $\{x_1^{(k)}, x_2^{(k)}, x_3^{(k)}\}, k=1,2,\cdots,n$,存在不同的函数 f_1、f_2、f_3、f_4,以及常量 C_1、C_2、B_1、B_2,有如下函数关系:

- 如果在固定 x_3 的情况下得出 x_1 和 x_2 的方程为

$$f_1(x_1) = C_1 f_2(x_2) + C_2 \tag{9.13}$$

在固定 x_2 的情况下得出 x_1 和 x_3 的方程为

$$f_1(x_1) = B_1 f_3(x_3) + B_2 \tag{9.14}$$

从严格意义上讲,在式(9.13)中常数 C_1、C_2 是 x_3 的函数;在式(9.14)中的常数 B_1、B_2 是 x_2 的函数。对于同一函数 $f_1(x_1)$ 应该有关于 x_2 和 x_3 的统一的公式,故对 $f_1(x_1)$ 而言,在式(9.13)中确定了 x_1 与 x_2 的关系,式(9.13)中确定了 x_1 与 x_3 的关系,合并式(9.13)、式(9.14),有如下启发式公式:

$$f_1(x_1) = C'_1 f_3(x_3) f_2(x_2) + C''_2 \tag{9.15}$$

$$f_1(x_1) = C'_1 f_2(x_2) + C'_2 f_3(x_3) + C''_3 \tag{9.16}$$

- 如果在固定 x_2 的情况下得出 x_1 和 x_3 的方程为:

$$f_3(x_1) = B'_1 f_4(x_3) + B'_2 \tag{9.17}$$

合并式(9.13)和式(9.17),则有如下多个启发式公式:

$$f_1(x_1) \theta f_3(x_1) = (C_1 f_2(x_2) + C_2) \theta (B'_1 f_4(x_3) + B'_2) \tag{9.18}$$

其中 θ 是+、-、*、/等操作。或者:

$$f_1(x_1) = g(x_1, x_2) + C_1 f_4(x_3) + C_2 f_2(x_2) + C_3 \tag{9.19}$$

g 函数的结构形式实质上是函数 f_2 和 f_3 的复合形式,由于 f_2 和 f_3 有系数项也有常数项,故 f_2 和 f_3 复合函数形式根据具体函数的不同有不同的合并方式,常见的是用一个公式的函数项去替换另外一个公式的系数和常数。

② 扩展到四维函数公式的启发式规则

设在三维数据的基础上增加一维数据 x_4，如果得到公式

$$f_2(x_2) = C_1 g(x_1, x_3) + C_2 \tag{9.20}$$

$$f_2(x_2) = C_3 f_4(x_4) + C_4 \tag{9.21}$$

则有如下启发式公式：

$$f_2(x_2) = C_1 g(x_1, x_3) f_4(x_4) + C_2 \tag{9.22}$$

$$f_2(x_2) = C_1 g(x_1, x_3) + C_2 f_4(x_4) + C_3' \tag{9.23}$$

③ 多维函数的扩展

通过增加函数变量的方法可以实现对多维函数变量公式的发现。多维函数扩展规则给出了函数公式的具体框架表示形式，最后必须通过给定的数据对各个启发式公式进行检验，决定公式的取舍。首先，通过实际给出的数据应用最小二乘法计算上式中各个常量的值；其次通过给定的数据确定各个启发式公式的误差，最后进行选择，满足误差需求的公式即为所求公式。

4. 三维函数公式的发现实例

(1) 试验数据

给定试验数据如表 9.12 所示。

表 9.12　三维数据实例

x_1	x_2	x_3	x_1	x_2	x_3
1.30	2.10	1.85	2.30	7.10	2.20
1.29	2.50	1.69	2.43	7.09	2.17
1.31	7.50	1.60	2.56	7.11	2.14
1.29	4.00	1.77	2.88	7.10	2.04
1.32	7.11	2.29	…	…	…

对于前 5 组数据，可以认为 x_1 为恒定，应用二维函数公式发现算法，找出变量 x_2 和 x_3 的关系，得到 5 个公式，选择误差最小的一个公式如下：

$$x_3^2 = 2.02\cos(x_2) + 4.46, \quad \text{误差为 } 0.0016 \tag{9.24}$$

对于后 5 组数据，可以认为 x_2 为恒定，应用二维函数公式发现算法，得到 3 个公式，选择误差最小的两个公式如下：

$$x_3^2 = 1.5\sin(x_1) + 7.75, \quad \text{误差为 } 0.00026 \tag{9.25}$$

$$\lg(x_3) = 0.07\sin(x_1) + 0.29, \quad \text{误差为 } 0.00015 \tag{9.26}$$

应用三维启发规则，将式(9.24)和式(9.25)合并，式(9.24)和式(9.26)合并，得到一系列公式，计算误差后得到满足误差要求的公式为

$$x_3^2 = 1.5\sin(x_1) + 2.02\cos(x_2) + 7.0 \tag{9.27}$$

该公式等式两端误差为 0.00041。

(2) 折射定律的发现

实验数据如表 9.13 所示(液体，温度为 20℃)。

表 9.13　不同介质间光线折射数据

物质	从空气中入射率 n_1(n_1，i 恒定)			从空气射入玻璃(n_1，n_2 恒定)	
	折射率 n_2	入射角 i	折射角 γ	入射角 i	折射角 γ
丙酮	1.3585	30	21.60	30	19.47
苯胺	1.5863	30	18.37	35	22.48
苯	1.5014	30	19.45	40	27.37
二硫化碳	1.6279	30	17.89	45	28.13
四氯化碳	1.4607	30	20.02	50	30.71
肉桂醛	1.6195	30	17.16	55	37.10
氯仿	1.4453	30	20.24	60	37.26
乙醇	1.3618	30	21.54		

设入射角为 i，折射角为 γ，入射线所在介质的折射率为 n_1，折射线所在介质的折射率为 n_2。因为光的可逆性，所以入射角和入射线的折射率与折射角和折射线的折射率两组数据可以互换，折射角 γ 改为入射角 i，入射角 i 变为折射角 γ，入射线和折射线所在位置的折射率也相应地调换。

对于从空气中入射到各介质，固定 n_1 和 i 角后，应用二维函数公式发现算法，得到折射率和折射角的公式

$$\sin(\gamma) = 0.5/n_2 \tag{9.28}$$

反之，从介质中入射到空气时(n_1 变为 n_2，i 角变为 γ 角)，固定 n_2 和 γ 角后，发现公式为

$$\sin(i) = 0.5/n_1 \tag{9.29}$$

在固定空气和玻璃两种介质时(n_1、n_2 恒定)，入射角 i 和折射角 γ 的关系，通过公式发现得：

$$\sin(i) = 1.5\sin(\gamma) \tag{9.30}$$

式(9.28)和式(9.29)两个公式从空气中入射不同物质的数据中生成，式(9.30)为从空气中入射玻璃的一组数据中生成。式(9.29)和式(9.30)应用三维扩展规则得：

$$\sin(i) = C_1\sin(\gamma)/n_1 + C_2, \quad 即\ \sin(\gamma) = C_1^* n_1 \sin(i) + C_2^* \tag{9.31}$$

对式(9.28)和式(9.31)利用四维扩展规则进行合并，得：

$$\sin(\gamma) = C_1''(n_1/n_2)\sin(i) + C_2'' \tag{9.32}$$

用已知的数据确定系数，得 $C_1''=1$，$C_2''=0$，即得 Snell 折射定律：

$$n_1\sin(i) = n_2\sin(\gamma) \tag{9.33}$$

5. FDD.1、FDD.2 和 FDD.3 的比较分析

FDD.2 通过引入导数规则对 FDD.1 算法的规则进行扩充，同时修改算法流程，使得算法运行更加合理，扩大了发现公式的宽度和广度。FDD.3 算法引入多维函数处理规则后对 FDD.2 算法进行了扩充，同时通过嵌套 FDD.2 算法流程，实现三维以上公式发现算法 FDD.3。把这 3 个进行比较分析，如表 9.14 所示。

表 9.14　FDD.1、FDD.2 和 FDD.3 的比较分析

比较方面	FDD.1	FDD.2	FDD.3
时间复杂度	$O(8nm)$	$O(2n^2m)$	$O(C_d^2 2n^2 m)$
流程循环	函数作用于一个变量	不同的函数作用于两个变量	
剪枝条件	误差最小原则	误差最小原则 误差收敛原则	误差最小原则 误差收敛原则
发现公式范围	初等函数、复合函数及其组合	在 FDD.1 基础上增加导数以及和导数相关的处理	在二维 FDD 基础上增加：三维扩展规则、多维扩展规则

注：n 为函数个数，m 为搜索树的深度，d 为维数。

在进行算法的时间复杂度分析时，由于搜索树的剪枝根据具体情况的不同而不同，所以假设在没有剪枝的情况下分析各个算法的时间复杂度。由于算法流程的不同，在发现同样形式的公式情况下，FDD.1 和 FDD.2、FDD.3 搜索树的深度不同，FDD.1 算法搜索树深度是 FDD.2、FDD.3 算法的两倍。

在 FDD.1 算法中，每个函数对两个变量分别作用的时间复杂度为 $O(2n)$，选择两个误差小的进入下面的分支，并且树的深度是 $2m$，则时间复杂度为 $O(8nm)$。

在 FDD.2 算法中，两个函数同时作用于两个变量的时间复杂度为 $O(nn)$，选择误差小的和误差收敛的进入下一个循环，则时间复杂度为 $O(2nnm)$。在 FDD.3 算法中，设函数的维数为 d，则任取其中的两个变量的组合为 C_d^2 个，所以整个算法的时间复杂度为 $O(C_d^2 2n^2 m)$。FDD.3 算法的发现公式的广度是以牺牲时间为代价的。

BACON 系统采用“项＝常数”的形式描述公式形式，而 FDD 采用“项＝初等函数或初等函数的复合形式”，并且引入导数规则等，和 BACON 相比，发现公式的范围和复杂度都有很大提高。

习　　题

1. 数据拟合的基本思想是什么？有什么优点和缺点？
2. 从 BACON 系统的实例看，公式发现与数据拟合有什么不同？
3. BACON 系统的简练算子有哪些？
4. BACON 系统是如何完成开普勒第三定律的发现的？
5. BACON 系统是如何发现理想气体定律的？
6. BACON 系统的启发式是什么？
7. 科学定律运用曲线拟合能发现吗？
8. FDD 系统的思想是什么？
9. FDD.1 系统的启发式函数是什么？
10. FDD.1 系统结构图的基本思想是什么？
11. FDD.1 系统中函数原型有哪些？

12．FDD.1 系统中的知识有哪些？

13．FDD.1 系统完成开普勒第三定律发现的过程是什么？它与 BACON 系统的发现过程有什么不同？

14．FDD.2 发现导数公式的启发式是什么？

15．FDD.3 发现多维函数公式的启发式是什么？

16．FDD 系统与 BACON 系统有什么不同？

第 10 章　神经网络与遗传算法

10.1　神经网络概念及几何意义

10.1.1　神经网络原理

1. 人工神经网络概念

神经生理学家和神经解剖学家早已证明，人的思维是通过人脑完成的，神经元是组成人脑的最基本单元，人脑神经元大约有 $10^{11} \sim 10^{12}$ 个（约 1000～10 000 亿个）。

神经元由细胞体、树突和轴突三部分组成，是一种根须状的曼延物。神经元的中心有一闭点，称为细胞体，它能对接收到的信息进行处理。细胞体周围的纤维有两类，轴突是较长的神经纤维，是发出信息的。树突的神经纤维较短，而分支很多，是接收信息的。一个神经元的轴突末端与另一个神经元的树突之间密切接触，传递神经元冲动的地方称为突触。经过突触的信息传递是有方向性的，不同的突触进行的冲动传递效果不一样，有的使后一神经元发生兴奋，有的使它受到抑制。每个神经元可有 $10 \sim 10^4$ 个突触。这表明大脑是一个广泛连接的复杂网络系统。从信息处理功能看，神经元具有如下性质：

（1）多输入单输出；

（2）突触兼有兴奋和抑制两种性能；

（3）可时间加权和空间加权；

（4）可产生脉冲；

（5）脉冲进行传递；

（6）非线性（有阈值）。

神经元的数学模型如图 10.1 所示。

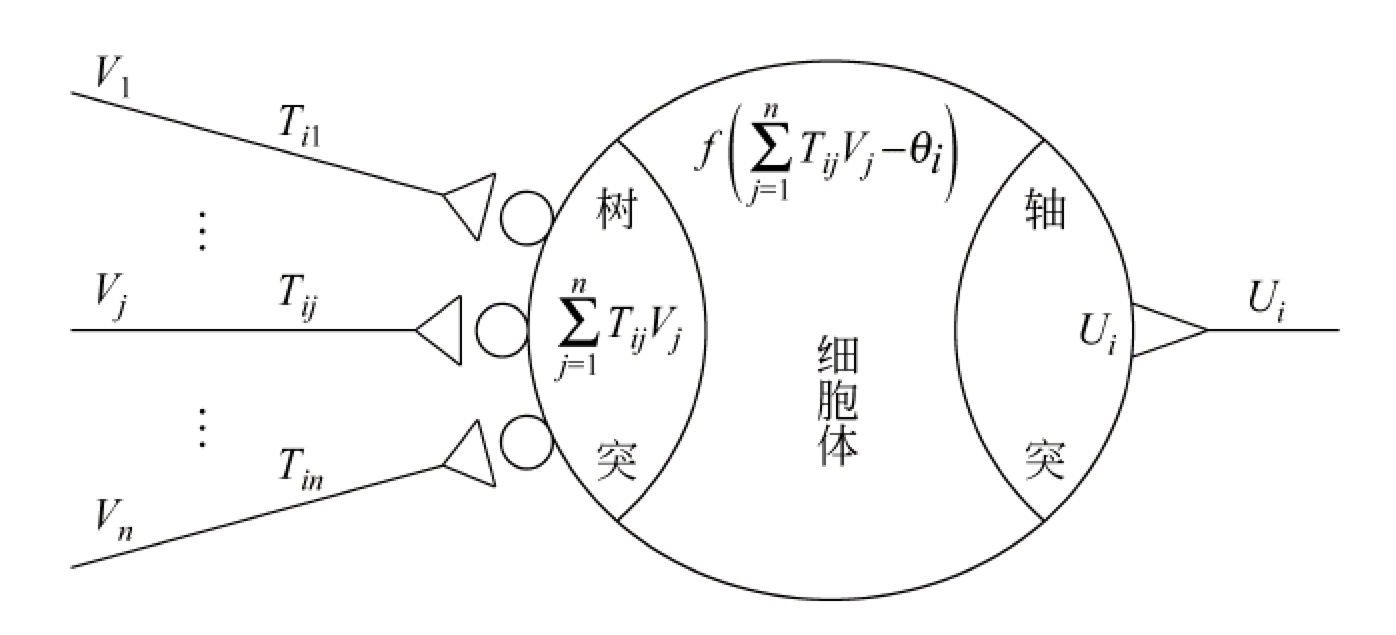


图 10.1　神经元模型

图 10.1 中 $V_1, V_2, \cdots, V_j, \cdots, V_n$ 为输入；U_i 为第 i 个神经元的输出；T_{ij} 为外面神经元与该神经元连接强度（即权），θ 为阈值，$f(X)$ 为该神经元的作用函数。

2. MP 模型与 Hebb 规则

(1) MP(Mcculloch 和 Pitts)模型

每个神经元的状态 $S_i(i=1,2,\cdots,n)$只取 0 或 1,分别代表抑制与兴奋。每个神经元的状态由 MP 方程决定:

$$S_i = f\left(\sum_j w_{ij} S_j - \theta_i\right), \quad i = 1,2,\cdots,n \tag{10.1}$$

其中 w_{ij}是神经元之间的连接强度,$w_{ii}=0$,$w_{ij}(i\neq j)$是可调实数,由学习过程来调整。θ_i是阈值,$f(x)$是阶梯函数。

(2) Hebb 规则

Hebb 学习规则:若 i 与 j 两种神经元之间同时处于兴奋状态,则它们间的连接应加强,即

$$\Delta w_{ij} = \alpha S_i S_j, \quad \alpha > 0 \tag{10.2}$$

这一规则与“条件反射”学说一致,并得到神经细胞学说的证实。设 $\alpha=1$,当 $S_i=S_j=1$ 时,$\Delta w_{ij}=1$,在 S_i、S_j中有一个为 0 时,$\Delta w_{ij}=0$。

3. 各种作用函数

(1) [0,1]阶梯函数

$$f(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leqslant 0 \end{cases} \tag{10.3}$$

(2) [−1,1]的阶梯函数

$$f(x) = \begin{cases} 1, & x > 0 \\ -1, & x \leqslant 0 \end{cases} \tag{10.4}$$

(3) (−1,1)S 型函数

$$f(x) = \frac{1-\mathrm{e}^{-x}}{1+\mathrm{e}^{-x}} \tag{10.5}$$

(4) (0,1)S 型函数(见图 10.2)

$$f(x) = \frac{1}{1+\mathrm{e}^{-x}} \tag{10.6}$$

图 10.2 (0,1)S 型函数

10.1.2 神经网络的几何意义

1. 神经元与超平面

由 n 个神经元$(j=1,2,\cdots,n)$对连接于神经元 i 的信息总输入 I_i为

$$I_i = \sum_{j=1}^{n} w_{ij} x_j - \theta_i \tag{10.7}$$

其中 w_{ij}为神经元j 到神经元i 的连接权值,θ_i为神经元的阈值。神经元 $x_j(j=1,2,\cdots,n)$相当于 n 维空间$(x_1,x_2,\cdots,x_n)$中一个结点的 n 维坐标(为了便于讨论,省略 i 下标记)。令

$$I = \sum_{j=1}^{n} w_j x_j - \theta = 0 \tag{10.8}$$

公式(10.8)代表了 n 维空间中以坐标 x_j 为变量的一个超平面。其中 w_j 为坐标的系数，θ 为常数项。

若已知有 n 个样本

$$(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)}), \quad k = 1, 2, \cdots, n$$

在 n 维空间中，相当于已知 n 个结点的各结点坐标。该 n 个结点可惟一构成一个超平面。超平面方程用行列式表示为

$$\begin{vmatrix} x_1 & x_2 & \cdots & x_n & 1 \\ x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} & 1 \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_1^{(n)} & x_2^{(n)} & \cdots & x_n^{(n)} & 1 \end{vmatrix} = 0 \tag{10.9}$$

式(10.9)是以 n 维坐标 $x_j(j=1,2,\cdots,n)$ 为变量的线性方程。将它展开即为超平面方程(10.8)。其中系数 w_j 和常数 θ 用行列式表示为

$$w_j = (-1)^{1+j} \begin{vmatrix} x_1^{(1)} & \cdots & x_{j-1}^{(1)} & x_{j+1}^{(1)} & \cdots & x_n^{(1)} & 1 \\ x_1^{(2)} & \cdots & x_{j-1}^{(2)} & x_{j+1}^{(2)} & \cdots & x_n^{(2)} & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_1^{(n)} & \cdots & x_{j-1}^{(n)} & x_{j+1}^{(n)} & \cdots & x_n^{(n)} & 1 \end{vmatrix} \tag{10.10}$$

$$-\theta = (-1)^n \begin{vmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ \cdots & \cdots & \cdots & \cdots \\ x_1^{(i)} & x_2^{(i)} & \cdots & x_n^{(i)} \\ \cdots & \cdots & \cdots & \cdots \\ x_1^{(n)} & x_2^{(n)} & \cdots & x_n^{(n)} \end{vmatrix} \tag{10.11}$$

当 $n=2$ 时，“超平面”为平面(x_1, x_2)上的一条直线：

$$I = \sum_{j=1}^{2} w_j x_j - \theta = w_1 x_1 + w_2 x_2 - \theta = 0$$

当 $n=3$ 时，“超平面”为空间(x_1, x_2, x_3)上的一个平面：

$$I = \sum_{j=1}^{3} w_j x_j - \theta = w_1 x_1 + w_2 x_2 + w_3 x_3 - \theta = 0$$

从几何角度看，一个神经元代表一个超平面。

2. 超平面的作用

n 维空间$(x_1, x_2, \cdots, x_n)$上的超平面 $I=0$，将空间划分为三部分。

(1) 平面本身

超平面上的任意结点$(x_1^{(0)}, x_2^{(0)}, \cdots, x_n^{(0)})$满足超平面方程，即

$$\sum_j w_j x_j^{(0)} - \theta = 0 \tag{10.12}$$

(2) 超平面上部 P

超平面上部 P 的任意结点$(x_1^{(p)},x_2^{(p)},\cdots,x_n^{(p)})$满足不等式,即

$$\sum_j w_j x_j^{(p)} - \theta > 0 \tag{10.13}$$

(3) 超平面下部 Q

超平面下部 Q 的任意结点$(x_1^{(q)},x_2^{(q)},\cdots,x_n^{(q)})$满足不等式,即

$$\sum_j w_j x_j^{(q)} - \theta < 0 \tag{10.14}$$

3. 作用函数的几何意义

神经网络中使用的阶梯型作用函数为

$$f(x)=\begin{cases}1, & x>0\\ 0, & x\leqslant 0\end{cases}$$

把 n 维空间中超平面的作用和神经网络作用函数结合起来,即

$$f(I)=f\left(\sum w_j x_j-\theta\right)=\begin{cases}1, & \sum_j w_j x_j-\theta>0\\ 0, & \sum_j w_j x_j-\theta\leqslant 0\end{cases} \tag{10.15}$$

式(10.15)的含义为:超平面上部 P 的任意结点经过作用函数后转换成数值 1。超平面下部 Q 上的任意结点经过作用函数后转换成数值 0。

4. 神经元的几何意义

通过以上分析可知,一个神经元将其他神经元对它的信息总输入 I,作用以后(通过作用函数)的输出,相当于该神经元所代表的超平面将 n 维空间(n 个输入神经元构成的空间)中超平面上部结点 P 转换成 1 类,超平面及其下部结点转换成 0 类。

结论:神经元起了一个分类作用。

5. 线性样本与非线性样本

定义 对空间中的一组两类样本,当能找出一个超平面将两者分开,称该样本是线性样本。若不能找到一个超平面将两者分开,则称该样本是非线性样本。

6. 非线性样本变换成线性样本

利用超平面分割空间原理,对一个非线性样本是不能用一个超平面分割开,但可用多个超平面分割空间成若干区,使每个区中只含同类样本的结点。这种分割完成了一种变换,使原非线性样本变换成二进制值下的新线性样本。

10.1.3 超曲面神经网络概念

超曲面神经网络是相对于超平面神经网络而言的。传统的神经网络是以 MP 模型为基础的,按 MP 模型,神经网络的公式为

$$s_i = f\left(\sum_j w_{ij} s_j - \theta_i\right) \quad i = 1,2,\cdots,n$$

其中每个神经网络元 s_i 代表了一个超平面(其中 s_j 是一次方):

$$I_i = \sum_j w_{ij} s_j - \theta_i = 0$$

神经网络的作用函数

$$f(I_i) = f\left(\sum_j w_{ij} s_j - \theta_i\right) = \begin{cases} 1, & \sum w_{ij} x_j - \theta_i > 0 \\ 0, & \sum w_{ij} x_j - \theta_i \leqslant 0 \end{cases}$$

相当于超平面 I_i 对 n 维空间进行了一次分割。多个超平面 $I_i(i=1,2,\cdots,n)$ 将 n 维空间进行了组合分割,把 n 维空间分成了若干个区域,使每个区域中只包含同类样本。这种区域分割完成了一次变换,即将非线性样本(不能用一个超平面分割的样本)通过多个超平面的分割使它变成了线性样本。对新的线性样本,再通过一次神经网络(超平面)就可完成对它的分割(分类)。BP 神经网络模型实质上就是通过两次超平面分割(即隐结点层和输出结点层)来完成样本分类的。

BP 神经网络是反复通过神经网络修改权值的迭代,最后找出隐结点神经网络超平面和输出结点神经网络超平面。

也可以用解析方法(非迭代方法)直接构造这些超平面来完成对各类样本的分割,已经有学者在从事这项工作并取得一些成果。

除了用超平面分割空间外,能否用超曲面分割空间实现对非线性样本的分割呢?这就要求神经网络公式(10.1)、(10.8)、(10.15)中 s_j 和 x_j 变成二次方以上。

黄金才提出的"超圆神经网络模型 CC"的公式为

$$y = f\left(\sum_i (x_i - a_i)^2 - c^2\right) \tag{10.16}$$

该神经网络模型与 MP 神经网络模型的比较,如图 10.3 所示。

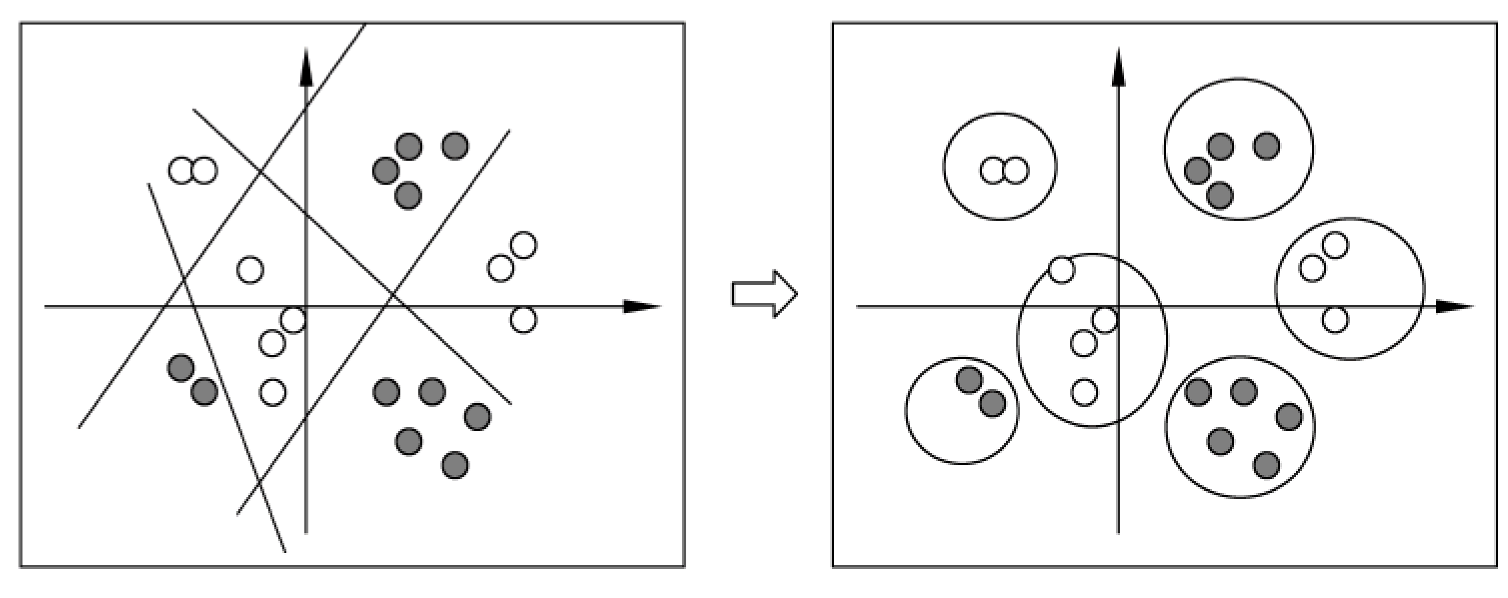


图 10.3　CC 模型与 MP 模型对样本空间的划分比较

黄金才还提出了"超曲面神经网络模型 Cover"的公式为

$$y = f(w_1 x + w_2 y + w_3 x^2 + w_4 xy + w_5 y^2 - c) \tag{10.17}$$

以上超曲面神经网络有效地达到了对非线性样本的分类效果。

超曲面神经网络是对神经网络的有益扩展。

10.2 感　知　机

10.2.1 感知机模型

1. 感知机(perceptron)原理

神经元 i 的输入为

$$I_i = \sum w_{ij} x_j - \theta_i$$

x_j为 j 神经元的输出,w_{ij}为神经元 j 到神经元 i 的连接权值。神经元 i 的输出为

$$O_i = f(I_i) \tag{10.18}$$

其中 $f(x)$为神经元作用函数,感知机采用[0,1]阶梯函数。

设 i 神经元的期望输出为 D_i,与计算输出 O_i之差为

$$\delta_i = D_i - O_i \tag{10.19}$$

通过样本学习,修正权值 w_{ij} 使 δ_i尽可能小。利用著名的德尔塔规则(delta rule)计算:

$$\Delta w_{ij} = \alpha \delta_i S_j \quad (\alpha \text{ 为常数}) \tag{10.20}$$

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij} \tag{10.21}$$

阈值修正公式:

$$\Delta \theta_i = \alpha \delta_i \tag{10.22}$$

$$\Delta \theta_i(t+1) = \theta_i(t) + \Delta \theta_i \tag{10.23}$$

更新权值 w_{ij} 和 θ_i。对样本重复以上计算,经过多次反复修正,将使 δ_i趋向于 0。

2. 感知机模型的实现

感知机是双层模型,如图 10.4 所示。

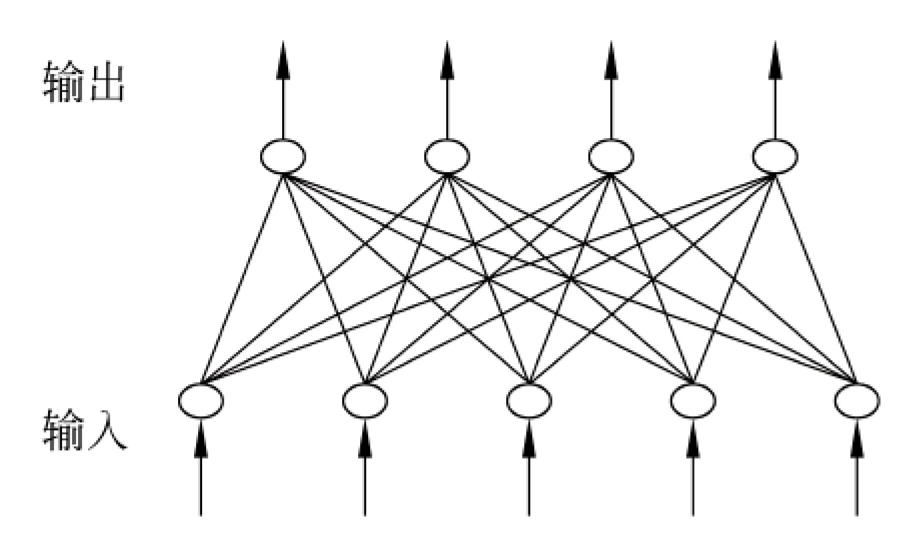


图 10.4　感知机网络结构

(1) 数据结构

① 输入结点(结点数为 m):$S_1, S_2, \cdots, S_m$

② 输出结点(结点数为 n)

结点:$1, 2, 3, \cdots, n$

输入:$I_1, I_2, I_3, \cdots, I_n$

输出:$O_1, O_2, O_3, \cdots, O_n$

③ 网络上权值

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{bmatrix}$$

(2) 学习过程(算法如下)

给出一组学习样本(共 p 个):$(S(1), D(1)), (S(2), D(2)), \cdots, (S(k), D(k)), \cdots, (S(p), D(p))$

对第 k 个样本 $(S(k),D(k))$ 有：

输入：$S(k)=(S_1(k),S_2(k),\cdots,S_m(k))$

期望输出：$D(k)=(D_1(k),D_2(k),\cdots,D_n(k))$

① 给网络上权值和阈值赋初值，如：$w_{ij}\equiv 0,\theta_i\equiv 0$。

样本循环变量赋初值：$k=1$，总误差初值 $E=0$，迭代次数 $L=0$。

② 通过感知机模型推理，对第 k 个样本：

输入：$S(k)=(S_1(k),S_2(k),\cdots,S_m(k))$

计算输出：$O(k)=(O_1(k),O_2(k),\cdots,O_n(k))$

迭代次数 L 加 1

③ 误差计算

每个输出结点误差：$\delta_i(k)=D_i(k)-O_i(k)$，　$(i=1,2,\cdots,n)$

第 k 个样本误差：$e_k=\sum_{i=1}^{n}|\delta_i(k)|$

④ 权值修正

原则：修正权 w_{ij} 使 δ_i 尽可能小，利用德尔塔规则(delta rule)，即

$$\Delta w_{ij}=\alpha\delta_i(k)\ S_j(k)$$

$$w_{ij}(n+1)=w_{ij}(n)+\Delta w_{ij}$$

⑤ 阈值修正

$$\Delta\theta_i=\alpha\delta_i$$

$$\Delta\theta_i(t+1)=\theta_i(t)+\Delta\theta_i$$

⑥ 计算 P 个样本的总误差 E 为

$$E=E+e_k$$

⑦ 检查：

$$k=P?$$

是：检查 $|E|\leqslant 0.05$?

　　是：计算结束。输出迭代次数 L 和总误差 E，输出网络权值 w_{ij}

　　否：$k=1,E=0$。样本再次学习，转②循环

否：k 加 1，做下一个样本，转②循环

10.2.2 感知机实例

两值逻辑加法例，输入数据和输出数据样本如下：

输入：$x1$	$x2$	输出：d(期望)
0	0	0
0	1	1
1	0	1
1	1	1

图 10.5　两值逻辑加法神经网络

该例的神经网络如图 10.5 所示。

该例的感知机计算公式：

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(k+1)}=\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(k)}+c(d-y)\begin{bmatrix} x1 \\ x2 \end{bmatrix}$$

初值：$\begin{bmatrix} w1 \\ w2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$， $c=1$

其中 d 为期望输出，y 为计算输出。

计算过程：

- $K=1$， $y=f(0+0)=0$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(1)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(0)} + (0-0)\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- $K=2$， $y=f(0+0)=0$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(2)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(1)} + (1-0)\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- $K=3$， $y=f(0+0)=0$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(3)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(2)} + (1-0)\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

- $K=4$， $y=f(1+1)=f(2)=1$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(4)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(3)} + (1-1)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

- 再循环一次，将会得到所有例子的$(d-y)$值均为零，即权值$(w1=1, w2=1)$满足所有实例要求。

二值逻辑加法样本示意如图 10.6 所示，两类样本(0,1)可以利用一条直线分隔开。

从线性样本定义可知二值逻辑加法是线性可分的。

感知机对线性样本是非常有效的，它在模式识别中是一个重要的方法。

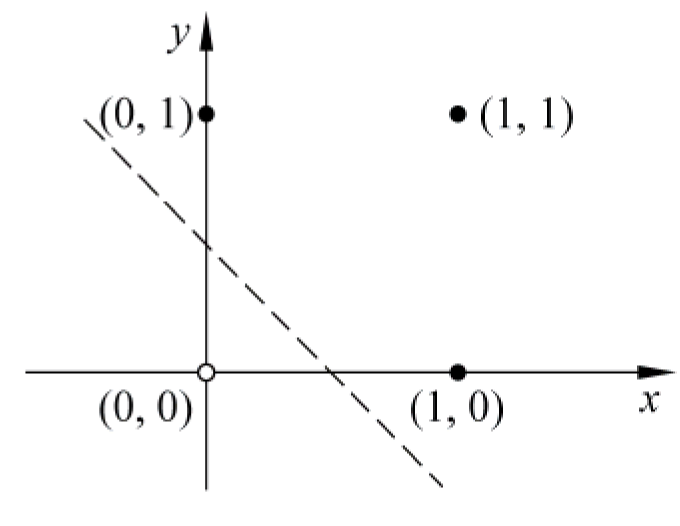


图 10.6　二值逻辑加法样本示意图

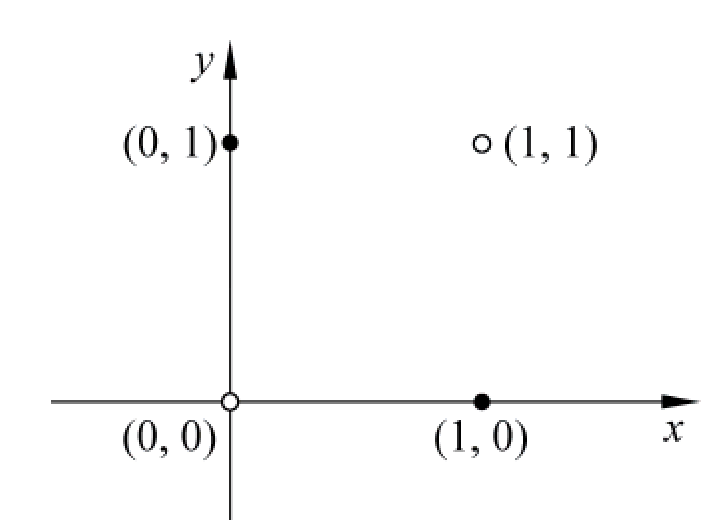


图 10.7　异或问题样本示意图

10.2.3　感知机讨论

现将二值逻辑加法例改为异或问题例，即第四个样本的输出值由 1 改为 0。异或问题样本示意如图 10.7 所示。

输入：$x1$	$x2$	输出：y
0	0	0
0	1	1
1	0	1
1	1	0

从图 10.7 可以看出，找不到一条直线将两类样本分开。从线性样本定义可知，该问题样本是一个非线性样本。

感知机对异或问题的神经网络计算如下：

$K=1,2,3$ 的计算同二值逻辑样本计算。

$K=4$ 时有：$y=f(1+1)=f(2)=1$。

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(4)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(3)} + (0-1)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

修改后的权值又回到了初始状态，如果继续计算，将出现无限循环，永远不会收敛。该例充分说明感知机对非线性样本无效。要解决非线性问题，需要在输入输出两层神经网络中间增加隐结点层。下面讨论的反向传播模型(BP)可以解决非线性问题。

10.3 反向传播模型

10.3.1 反向传播模型 BP 网络结构

BP(back propagation)模型是 1985 年由 Rumelhart 等人提出的。

1. 多层网络结构

神经网络不仅有输入结点、输出结点，而且有一层或多层隐结点，如图 10.8 所示。

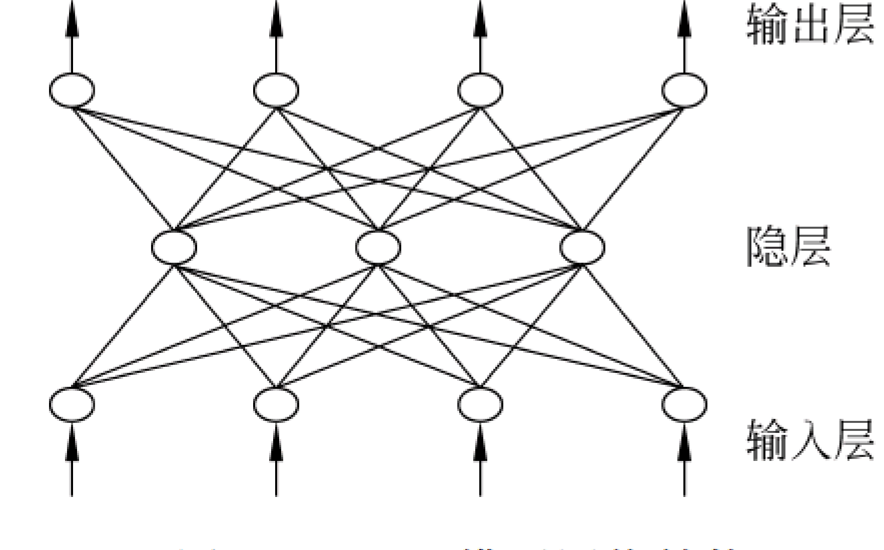


图 10.8 BP 模型网络结构

2. 作用函数为(0,1)S 型函数

$$f(x)=\frac{1}{1+e^{-x}} \tag{10.24}$$

3. 误差函数

对第 p 个样本误差计算公式为

$$E_p=\frac{1}{2}\sum_i (t_{pi}-O_{pi})^2 \tag{10.25}$$

其中 t_{pi}、O_{pi} 分别是期望输出与计算输出。

10.3.2 BP 网络学习公式推导

BP 网络表示：输入结点为 x_j，隐结点为 y_i，输出结点为 O_l。

输入结点与隐结点间的网络权值为 w_{ij}，隐结点与输出结点间的网络权值为 T_{li}。当输出结点的期望输出为 t_l时，BP 模型的计算公式如下。

1. 隐结点的输出

$$y_i=f\left(\sum_j w_{ij}x_j-\theta_i\right)=f(\mathrm{net}_i)$$

其中 $net_i = \sum_j w_{ij} x_j - \theta_i$。

2. 输出结点计算输出

$$O_l = f\left(\sum_i T_{li} y_i - \theta_l\right) = f(net_l)$$

其中 $net_l = \sum_i T_{li} y_i - \theta_l$。

3. 输出结点的误差公式

$$\begin{aligned} E &= \frac{1}{2}\sum_l (t_l - O_l)^2 = \frac{1}{2}\sum_l \left(t_l - f\left(\sum_i T_{li} y_i - \theta_l\right)\right)^2 \\ &= \frac{1}{2}\sum_l \left(t_l - f\left(\sum_i T_{li} f\left(\sum_j w_{ij} x_j - \theta_i\right) - \theta_l\right)\right)^2 \end{aligned}$$

（1）对输出结点的公式推导

$$\frac{\partial E}{\partial T_{li}} = \sum_{k=1}^{n} \frac{\partial E}{\partial O_k} \frac{\partial O_k}{\partial T_{li}} = \frac{\partial E}{\partial O_l} \frac{\partial O_l}{\partial T_{li}}$$

其中 E 是多个 O_k 的函数，但只有一个 O_l 与 T_{li} 有关，各 O_k 间相互独立。其中

$$\frac{\partial E}{\partial O_l} = \frac{1}{2}\sum_k -2(t_k - O_k)\frac{\partial O_k}{\partial O_l} = -(t_l - O_l)$$

$$\frac{\partial O_l}{\partial T_{li}} = \frac{\partial O_l}{\partial net_l} \frac{\partial net_l}{\partial T_{li}} = f'(net_l) y_i$$

则

$$\frac{\partial E}{\partial T_{li}} = -(t_l - O_l) f'(net_l) y_i \tag{10.26}$$

设输出结点误差

$$\delta_l = (t_l - O_l) f'(net_l) \tag{10.27}$$

则

$$\frac{\partial E}{\partial T_{li}} = -\delta_l y_i \tag{10.28}$$

（2）对隐结点的公式推导

$$\frac{\partial E}{\partial w_{ij}} = \sum_l \sum_i \frac{\partial E}{\partial O_l} \frac{\partial O_l}{\partial y_i} \frac{\partial y_i}{\partial w_{ij}}$$

其中 E 是多个 O_l 函数，针对某一个 w_{ij}，对应一个 y_i，它与所有 O_l 有关，其中：

$$\frac{\partial E}{\partial O_l} = \frac{1}{2}\sum_k -2(t_k - O_k)\frac{\partial O_k}{\partial O_l} = -(t_l - O_l)$$

$$\frac{\partial O_l}{\partial y_i} = \frac{\partial O_l}{\partial net_l} \frac{\partial net_l}{\partial y_i} = f'(net_l) \frac{\partial net_l}{\partial y_i} = f'(net_l) T_{li}$$

$$\frac{\partial y_i}{\partial w_{ij}} = \frac{\partial y}{\partial net_i} \frac{\partial net_i}{\partial w_{ij}} = f'(net_i) x_j$$

则

$$\frac{\partial E}{\partial w_{ij}}=-\sum_l(t_l-O_l)f'(\mathrm{net}_l)T_{li}f'(\mathrm{net}_i)x_j=-\sum_l\delta_l T_{li}f'(\mathrm{net}_i)x_j \tag{10.29}$$

设隐结点误差

$$\delta_i'=f'(\mathrm{net}_i)\sum_l\delta_l T_{li} \tag{10.30}$$

则

$$\frac{\partial E}{\partial w_{ij}}=-\delta_i' x_j \tag{10.31}$$

由于权值的修正 ΔT_{li} 和 Δw_{ij} 正比于误差函数沿梯度下降，有

$$\Delta T_{li}=-\eta\frac{\partial E}{\partial T_{li}}=\eta\delta_l y_i \tag{10.32}$$

$$\delta_l=(t_l-O_l)f'(\mathrm{net}_l) \tag{10.27}$$

$$\Delta w_{ij}=-\eta'\frac{\partial E}{\partial w_{ij}}=\eta'\delta_i x_j \tag{10.33}$$

$$\delta_i'=f'(\mathrm{net}_i)\sum_l\delta_l T_{li} \tag{10.30}$$

(3) 基本公式汇总

① 对输出结点误差：

$$\delta_l=(t_l-O_l)f'(\mathrm{net}_l) \tag{10.34}$$

② 输出层网络权值修正：

$$T_{li}(k+1)=T_{li}(k)+\Delta T_{li}=T_{li}(k)+\eta\delta_l y_i \tag{10.35}$$

③ 对隐结点误差：

$$\delta_i'=f'(\mathrm{net}_i)\sum_l\delta_l T_{li} \tag{10.36}$$

④ 隐结点网络权值修正：

$$w_{ij}(k+1)=w_{ij}(k)+\Delta w_{ij}=w_{ij}(k)+\eta'\delta_i' x_j \tag{10.37}$$

其中，隐结点误差 δ_i' 的含义：$\sum\limits_l\delta_l T_{li}$ 表示输出层结点 l 的误差 δ_l 通过权值 T_{li} 向隐结点 i 反向传播（误差 δ_l 乘权值 T_{li} 再累加）成为隐结点 i 的误差，如图 10.9 所示。

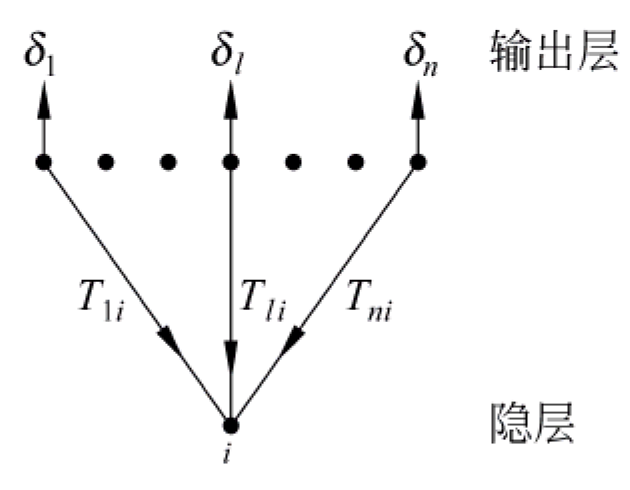


图 10.9 误差反向传播示意图

4. 阈值的修正

阈值 θ 也是一个变化值，在修正权值的同时也修正它，原理同权值的修正。

(1) 对输出结点的公式推导

$$\frac{\partial E}{\partial\theta_l}=\frac{\partial E}{\partial O_l}\frac{\partial O_l}{\partial\theta_l}$$

其中 $\frac{\partial E}{\partial O_l}=-(t_l-O_l)$，对某个 θ_l 对应一个 O_l。

$$\frac{\partial O_l}{\partial\theta_l}=\frac{\partial O_l}{\partial \mathrm{net}_l}\frac{\partial \mathrm{net}_l}{\partial\theta_l}=f'(\mathrm{net}_l)(-1)$$

则

$$\frac{\partial E}{\partial \theta_l} = (t_l - O_l) f'(\text{net}_l) = \delta_l \tag{10.38}$$

由于

$$\Delta\theta_l = \eta \frac{\partial E}{\partial \theta_l} = \eta\delta_l$$

则

$$\theta_l(k+1) = \theta_l(k) + \eta\delta_l \tag{10.39}$$

（2）对隐结点的公式推导

$$\frac{\partial E}{\partial \theta_i} = \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial \theta_i} = \frac{\partial E}{\partial O_l}\frac{\partial O_l}{\partial y_i}\frac{\partial y_i}{\partial \theta_i}$$

其中

$$\frac{\partial E}{\partial O_l} = -\sum_l (t_l - O_l)$$

$$\frac{\partial O_l}{\partial y_i} = f'(\text{net}_l) T_{li}$$

$$\frac{\partial y_i}{\partial \theta_i} = \frac{\partial y}{\partial \text{net}_l}\frac{\partial \text{net}_l}{\partial \theta_i} = f'(\text{net}_i)(-1) = -f'(\text{net}_i)$$

则

$$\frac{\partial E}{\partial \theta_i} = \sum_l (t_l - O_l) f(\text{net}_l) T_{li} f(\text{net}_i) = \sum_l \delta_l T_{li} f(\text{net}_i) = \delta'_i \tag{10.40}$$

由于

$$\Delta\theta_i = \eta \frac{\partial E}{\partial \theta_i} = \eta'\delta'_i$$

则

$$\theta_i(k+1) = \theta_i(k) + \eta'\delta'_i \tag{10.41}$$

5. 作用函数 $f(x)$ 的导数公式

函数 $f(x)=\frac{1}{1+\mathrm{e}^{-x}}$，存在关系 $f'(x)=f(x)(1-f(x))$

则

$$f'(\text{net}_k) = f(\text{net}_k)(1 - f(\text{net}_k)) \tag{10.42}$$

对输出结点

$$O_l = f(\text{net}_l)$$

$$f'(\text{net}_l) = O_l(1 - O_l) \tag{10.43}$$

对隐结点

$$y_i = f(\text{net}_i)$$

$$f'(\text{net}_i) = y_i(1 - y_i) \tag{10.44}$$

6. BP 模型计算公式汇总

（1）输出结点输出 O_l 的计算公式

① 输入结点的输入：x_j

② 隐结点的输出：$y_i = f\left(\sum_j w_{ij} x_j - \theta_i\right)$，其中：连接权值为 w_{ij}，结点阈值为 θ_i。

③ 输出结点输出：$O_l = f\left(\sum_i T_{li} y_i - \theta_l\right)$，其中：连接权值为 T_{li}，结点阈值为 θ_l。

(2) 输出层(隐结点到输出结点间)的修正公式

① 输出结点的期望输出：t_l

② 误差控制：

所有样本误差：$E = \sum_{k=1}^{P} e_k < \varepsilon$，其中一个样本误差 $e_k = \sum_{l=1}^{n} \mid t_l^{(k)} - O_l^{(k)} \mid$，其中，$p$ 为样本数，n 为输出结点数。

③ 误差公式：

$$\delta_l = (t_l - O_l) O_l (1 - O_l) \tag{10.45}$$

④ 权值修正：

$$T_{li}(k+1) = T_{li}(k) + \eta \delta_l y_i, \quad \text{其中 } k \text{ 为迭代次数。} \tag{10.46}$$

⑤ 阈值修正：

$$\theta_l(k+1) = \theta_l(k) + \eta \delta_l \tag{10.47}$$

(3) 隐结点层(输入结点到隐结点间)的修正公式

① 误差公式：

$$\delta'_i = y_i (1 - y_i) \sum_l \delta_l T_{li} \tag{10.48}$$

② 权值修正：

$$w_{ij}(k+1) = w_{ij}(k) + \eta' \delta'_i x_j \tag{10.49}$$

③ 阈值修正：

$$\theta_i(k+1) = \theta_i(k) + \eta' \delta'_i \tag{10.50}$$

BP 模型算法分为 3 部分：

① 隐结点和输出结点的输出计算；

② 输出结点和隐结点的误差计算；

③ 输出层网络权值及结点阈值与隐结点层网络权值及结点阈值的修改，如图 10.10 所示。

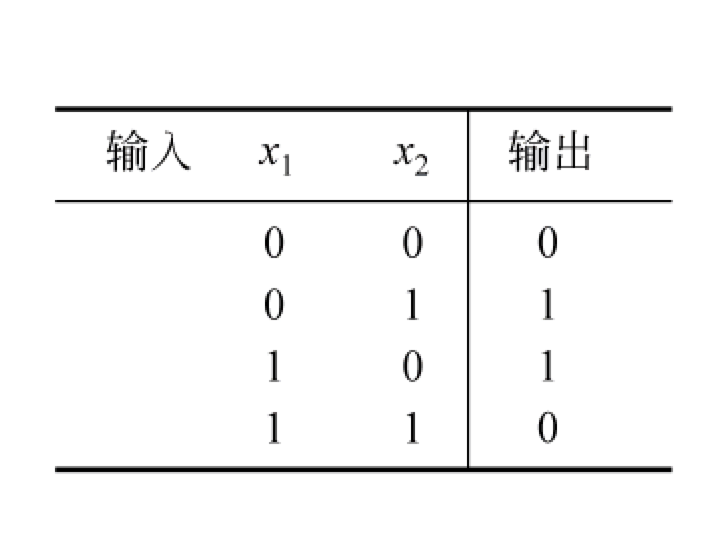
图 10.10 BP 模型算法示意图

BP 模型计算,不但对每一个样本要积累计算各输出结点的误差,对所有样本还要积累各样本的误差,这个总误差才是一次迭代的误差,当它不满足给定误差时,继续迭代(用新网络权值和阈值,再对所有样本重复计算),直到满足给定误差为止。这种迭代可能要上万次才能够收敛。

10.3.3 实例分析

1. 异或问题的 BP 神经网络

异或问题(XOR)用 BP 模型进行求解,样本和神经网络如图 10.11 所示。



输入	x_1	x_2	输出
	0	0	0
	0	1	1
	1	0	1
	1	1	0

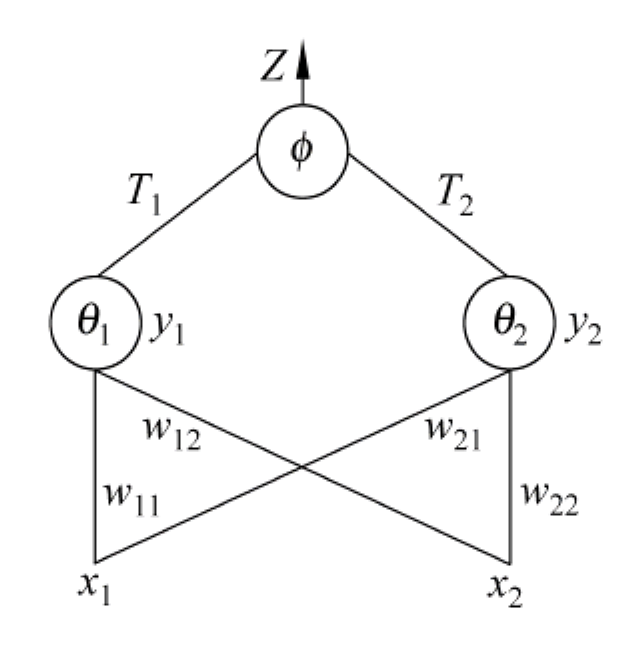


图 10.11　异或问题神经网络图

按问题要求,设置输入结点 2 个(x_1,x_2),输出结点 1 个(Z),隐结点 2 个(y_1,y_2)。

2. 计算机运行结果

(1) 迭代次数:16 745 次;给定误差:0.05。

(2) 隐层网络权值和阈值:$w_{11}=5.24$,$w_{12}=5.23$,$w_{21}=6.68$,$w_{22}=6.64$,$\theta_1=8.01$,$\theta_2=2.98$。

(3) 输出层网络权值和阈值:$T_1=-10$,$T_2=10$,$\phi=4.79$。

3. 用计算结果分析神经网络的几何意义

(1) 隐结点代表的直线方程(见图 10.12)

$$y_1: 5.24x_1+5.23x_2-8.01=0$$

即

$$x_1+0.998x_2-1.529=0 \tag{10.51}$$

$$y_2: 6.68x_1+6.64x_2-2.98=0$$

即

$$x_1+0.994x_2-0.446=0 \tag{10.52}$$

① 直线 y_1 和 y_2 将平面(x_1,x_2)分为 3 个区:

- y_1 线上方区,$x_1+x_2-1.53>0$,$x_1+x_2-0.45>0$
- y_1、y_2 线之间区,$x_1+x_2-1.53<0$,$x_1+x_2-0.45>0$
- y_2 线的下方区,$x_1+x_2-1.53<0$,$x_1+x_2-0.45<0$

② 对样本点：

- 点(0,0)落入 y_2 的下方区，经过隐结点作用函数 $f(x)$（暂取它为阶梯函数），得到输出 $y_1=0, y_2=0$。
- 点(1,0)和点(0,1)落入 y_1、y_2 线之间区，经过隐结点作用函数 $f(x)$，得到输出均为 $y_1=0, y_2=1$。
- 点(1,1)落入 y_1 线上方区，经过隐结点作用函数 $f(x)$，得到输出为 $y_1=1, y_2=1$。

③ 结论：隐结点将 x_1、x_2 平面上 4 个样本点(0,0)、(0,1)、(1,0)、(1,1)变换成 3 个样本点(0,0)、(0,1)、(1,1)，它已是线性样本。

(2) 输出结点代表的直线方程(见图 10.13)

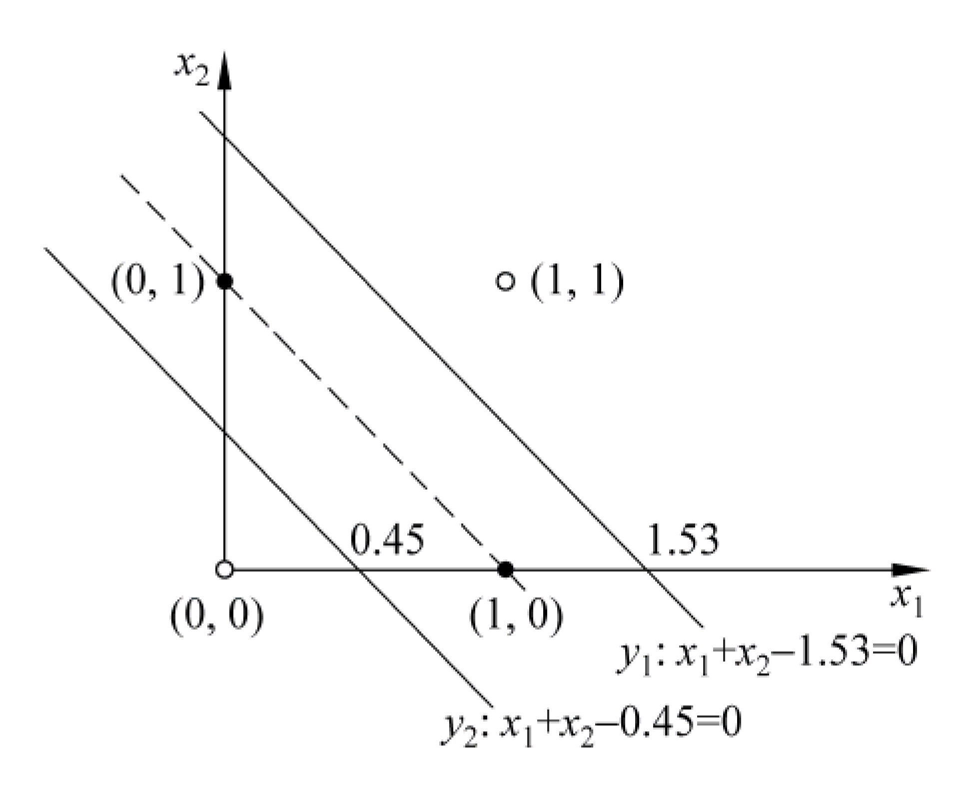


图 10.12　隐结点代表的直线方程

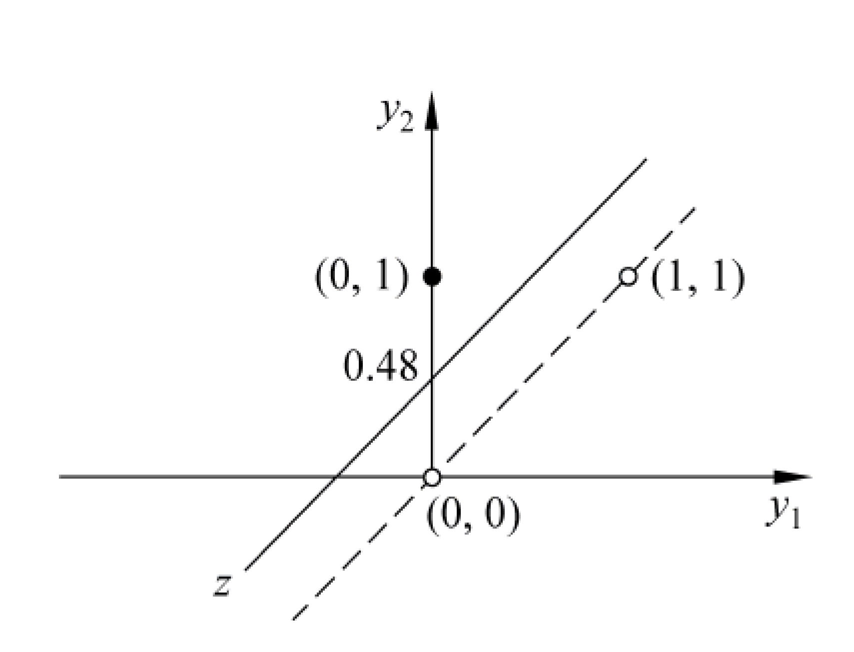


图 10.13　输出结点代表的直线方程

$$Z: -10y_1+10y_2-4.79=0$$

即

$$-y_1+y_2-0.479=0 \tag{10.53}$$

① 直线 Z 将平面 (y_1, y_2) 分为两个区：

- Z 线上方区，$-y_1+y_2-0.479>0$
- Z 线下方区，$-y_1+y_2-0.479<0$

② 对样本点：

- 点(0,1)(即 $y_1=0, y_2=1$)落入 Z 线上方区，经过输出结点作用函数 $f(x)$（暂取它为阶梯函数）得到输出为：$Z=1$。
- 点(0,0)(即 $y_1=0, y_2=0$)，点(1,1)(即 $y_1=1, y_2=1$)落入 Z 线下方区，经过输出结点作用函数 $f(x)$ 得到输出为：$Z=0$。

③ 结论：输出结点将 y_1、y_2 平面上 3 个样本(0,0)、(0,1)、(1,1)变换成两类样本 $Z=1$ 和 $Z=0$。

4. 神经网络结点的作用

从上面的分析中可以得出结论：

(1) 隐结点的作用是将原非线性样本(4 个)变换成线性样本(3 个)。

(2) 输出结点的作用是将线性样本(3 个)变换成两类(1 类或 0 类)。

对于作用函数 $f(x)$取为 S 型函数，最后变换成两类为“接近 1 类”和“接近 0 类”。

5. 超平面(直线)特性

(1) 隐结点直线特性

隐结点直线 y_1、y_2 相互平行，且平行于过点(1,0)和点(0,1)的直线 L：$x_1+x_2-1=0$。直线 y_1 位于点(1,1)到直线 L 的中间位置附近($\theta_1=1.53$)。直线 y_2 位于点(0,0)到直线 L 的中间位置附近($\theta_2=0.45$)。阈值 θ_1 和 θ_2 可以在一定范围内变化：$1.0\leqslant\theta_1<2$，$0\leqslant\theta_2<1.0$。其分类效果是相同的。

(2) 输出结点直线特性

输出结点直线 Z，平行于过点(0,0)和点(1,1)的直线 P：$y_1-y_2=0$。直线 Z 位于点(0,1)到直线 P 的中间位置附近($\phi=0.48$)。阈值 ϕ 可以在一定范围内变化($0\leqslant\phi<1$)，其分类效果是相同的。

10.4 遗 传 算 法

遗传算法是模拟生物进化的自然选择和遗传机制的一种寻优算法。它模拟了生物的繁殖、交配和变异现象，从任意一初始种群出发，产生一群新的更适应环境的后代。这样一代一代不断繁殖、进化，最后收敛到一个最适应环境的个体上。遗传算法对于复杂的优化问题无需建立数学模型和进行复杂运算，只需要利用遗传算法的算子就能寻找到问题的最优解或满意解。

自然选择学说认为，生物要生存下去，就必须进行生存斗争。生存斗争包括种内斗争、种间斗争以及生物跟环境之间的斗争 3 个方面。在生存斗争中，具有有利变异的个体容易存活下来，并且有更多的机会将有利变异传给后代；具有不利变异的个体就容易被淘汰，产生后代的机会也少得多。因此，凡是在生存斗争中获胜的个体都是对环境适应性比较强的。达尔文把这种在生存斗争中适者生存、不适者淘汰的过程叫做自然选择。达尔文的自然选择学说表明，遗传和变异是决定生物进化的内在因素。遗传是指父代与子代之间，在性状上存在的相似现象。变异是指父代与子代之间，以及子代个体之间，在性状上或多或少地存在的差异现象。在生物体内，遗传和变异的关系十分密切。一个生物体的遗传性状往往会发生变异，而变异的性状有的可以遗传。遗传能使生物的性状不断地传送给后代，因此保持了物种的特性，变异能够使生物的性状发生改变，从而适应新的环境而不断地向前发展。

生物的遗传与变异有它的物质基础。遗传物质的主要载体是染色体(chromsome)。染色体主要是由 DNA(脱氧核糖核酸)和蛋白质组成。基因(gene)是染色体的片段，它储存着遗传信息，可以准确地复制，也能够发生突变，生物体自身通过对基因的复制(reproduction)和交叉(crossover，即基因自由组合和基因连锁互换)的操作实现性状的遗传。同时，通过基因变异实现生物性状的变异。根据达尔文进化论，多种多样的生物之所以能够适应环境而得以生存进化，是和上述的遗传和变异生命现象分不开的。生物的遗传特性使生物界的物种能够保持相对的稳定；生物的变异特性使生物个体产生新的性状，以至于形成了新的

物种，推动了生物的进化和发展。

10.4.1　遗传算法基本原理

1. 概述

遗传算法(genetic algorithms，GA)是一种基于遗传学的搜索优化算法。遗传学认为遗传是作为一种指令码封装在每个染色体个体中，并以基因(位)的形式包含在染色体(个体)中。每个基因有特殊的位置并控制某个特殊的性质。由基因组成的个体对环境有一定的适应性。基因杂交和基因突变能产生对环境适应性强的后代，通过优胜劣汰的自然选择，适应值高的基因结构就保存下来。

在遗传算法中，“染色体”对应的是数据或数组，通常是由一维的串结构数据来表现。串上各个位置对应“基因”，而各位置上的值对应基因的取值。基因组成的串就是染色体，或者叫做基因型个体(individuals)。一定数量的个体组成了群体(population)。群体中个体的数目称为群体的大小(population size)，也叫群体规模。而各个体对环境的适应程度叫做适应度(fitness)。

遗传算法中包含两个必需的数据转换操作，一个是把搜索空间中的参数或解转换成遗传空间中的染色体或个体，此过程又叫做编码(coding)操作；另一个是相反操作，叫做译码(decoding)操作。

遗传算法是一种群体型操作，该操作以群体中的所有个体为对象。选择(selection)、交叉(crossover)和变异(mutation)是遗传算法的3个主要操作算子，它们构成了遗传操作(genetic operation)，使遗传算法具有了其他传统方法所没有的特性。

遗传算法的处理流程如图10.14所示。

遗传算法首先将问题的每个可能的解按某种形式进行编码，编码后的解称做染色体(个体)。随机选取 N 个染色体构成初始种群，再根据预定的评价函数对每个染色体计算适应值，使得性能较好的染色体具有较高的适应值。选择适应值高的染色体进行复制，通过遗传算子进行选择、交叉(重组)、变异，来产生一群新的更适应环境的染色体，形成新的种群。这样一代一代不断繁殖、进化，最后收敛到一个最适应环境的个体上，求得问题的最优解。

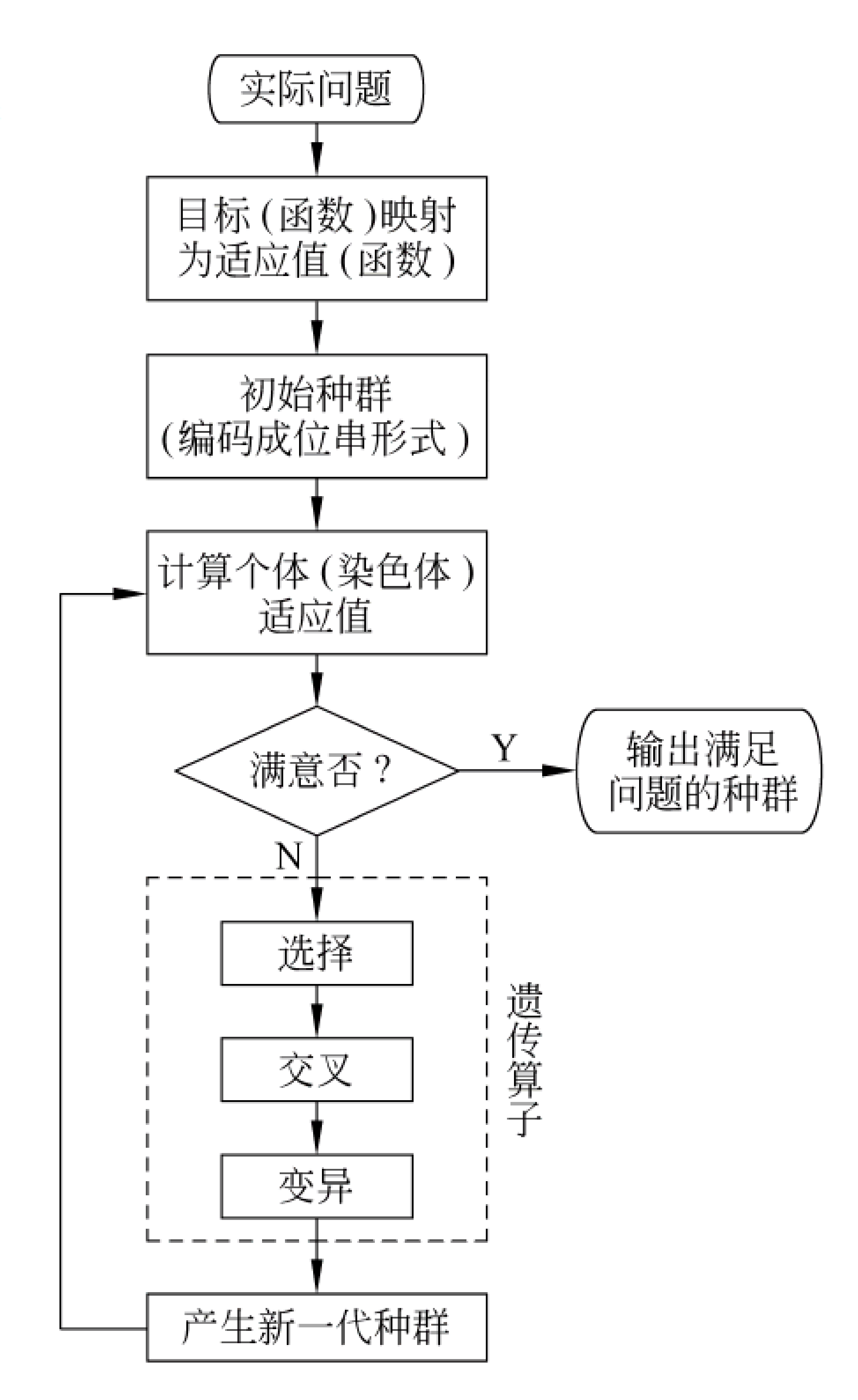


图10.14　遗传算法的处理流程示意图

2. 遗传算法中的基本要素

遗传算法中包含了如下5个基本要素：问题编码；初始群体的设定；适应值函数的设计；遗传

操作设计；控制参数设定（主要是指群体大小和使用遗传操作的概率等）。这 5 个要素构成了遗传算法的核心内容。

（1）问题编码

将子串拼接起来构成“染色体”位串。但是不同串长和不同的码制，对问题求解的精度和遗传算法收敛时间会有很多影响。如何将问题描述成串的形式就不那么简单，而且同一问题可以有不同的编码方法。

常用的二进制编码方式是基于确定的二进制位串上：$I=\{0,1\}^L$。目前也出现采用其他编码方式，如用向量（向量元素为实数）来表示染色体，或者用规则形式（规则 A，规则 B，规则 C……）来表示染色体。

（2）初始群体的生成

遗传算法是群体型操作，这样必须为遗传操作准备一个由若干初始解组成的初始群体。初始群体的每个个体都是通过随机方法产生的。初始群体也称为进化的初始代，即第一代（first generation）。

（3）适应值函数的确定

遗传算法在搜索进化过程中一般不需要其他外部信息，仅用评估函数值来评估个体或解的优劣，并作为以后遗传操作的依据。评估函数值又称做适应值（fitness）。

适应值函数（即评估函数）是根据目标函数确定的。适应值总是非负的，任何情况下总是希望越大越好。一般目标函数有正有负，且和适应值之间的关系也是多种多样的。如求最大值时，目标函数与适应值变化方向一致，而求最小值时，变化方向正好相反。因此，存在目标函数到适应值函数的映射问题，常见的映射形式为

$$\phi(\alpha) = \delta(f(\tau(\alpha)))$$

其中，α 为个体；$\tau(\alpha)$为个体的译码函数；f 则为具体求解问题的表达式；δ 为变换函数，δ 的作用是确保适应值为正，并且最好的个体其适应值最大。适应值函数的选取至关重要，它直接影响到算法的收敛速度，即最终能否找到最优解。函数优化问题可直接将函数本身作为评价函数。而对于复杂系统的评价函数一般不那么直观，往往需要研究者自己构造出能对解的性能进行评价的函数。

为了使遗传算法有效地工作，必须保持种群内位串的多样性和位串之间的竞争机制。如果将遗传算法的运行分为开始、中间和结束 3 个阶段，在开始阶段中，若一个规模不太大的种群内有少数非凡的个体（适应值很高的位串）的话，按通常的选择方法，这些个体会被大量繁殖，在种群中占有大的比重，这样就会减少种群的多样性，导致过早收敛，从而可能丢失一些有意义的搜索点或最优点，而陷入局部最优。其次，在结束阶段，即使种群内保持了很大的多样性，但若所有或大多数个体都有很高的适应值，从而种群平均适应值和最大适应值相差无几，那么平均适应值附近的个体和具有最高适应值的个体，被选中的机会几乎相同，这样选择就成了一个近乎随机的步骤，适应值的作用就会消失，从而搜索性能得不到明显改进。因此，有必要对种群内各位串的适应值进行有效地调整，既不能相差太大，又要拉开档次，强化位串之间的竞争性。最常见的调整方法是线性调整法。

10.4.2 遗传算子

遗传算法的执行过程中,每一代有许多不同的染色体(个体)同时存在,这些染色体中哪个保留(生存)、哪个淘汰(死亡)是根据它们对环境的适应能力决定的,适应性强的有更多的机会保留下来。适应性强弱是计算个体适应值函数 $f(x)$的值来判别的,这个值称为适应值(fitness)。适应值函数 $f(x)$的构成与目标函数有密切关系,往往是目标函数的变种。主要的遗传算子有如下几种。

1. 选择(Selection)算子

它又称复制(reproduction)、繁殖算子。

选择是从种群中选择生命力强的染色体产生新种群的过程。依据每个染色体的适应值大小,适应值越大,被选中的概率就越大,其子孙在下一代产生的个数就越多。

选择操作是建立在群体中个体的适应值评估基础上的,目前常用的选择算子有以下几种。

(1) 适应值比例法

适应值比例法是目前遗传算法中最常用的选择方法。它也叫赌轮或蒙特卡罗(Monte Carlo)选择。在该方法中,各个个体的选择概率和其适应值成比例。

设群体大小为 n,其中个体 i 的适应值为 f_i,则 i 被选择的概率 P_i为

$$P_i = f_i \Big/ \sum_{j=1}^{M} f_j \tag{10.54}$$

显然,概率 P_i反映了个体 i 的适应值在整个群体的个体适应值总和中所占的比例。个体适应值越大,其被选择的概率就越高。按式(10.54)计算出群体中各个个体的选择概率后,就可以决定哪些个体被选出。

(2) 最佳个体保存法

该方法的思想是把群体中适应度最高的个体不进行配对交叉而直接复制到下一代中。此种选择操作又称复制(copy)。

设在第 t 代中,群体中 $a^*(t)$为最佳个体。而在 $A(t+1)$新一代群体中不存在 $a^*(t)$,则把 $a^*(t)$作为 $A(t+1)$中的第 $n+1$ 个个体(其中 n 为群体大小)。

采用此选择方法的优点是,进化过程中某一代的最优解可不被交叉和变异操作破坏,但是,会使进化有可能限于局部解,即它更适合单峰性质的空间搜索。一般它都与其他选择方法结合使用。

(3) 期望值方法

① 计算群体中每个个体在下一代生存的期望数目:

$$M = f_i/\bar{f} = f_i \Big/ \sum f_i/n \tag{10.55}$$

② 若某个体被选中并要参与配对和交叉,则它在下一代中的生存的期望数目减去 0.5;若不参与配对和交叉,则该个体的生存期望数目减去 1。

③ 在②的两种情况中,若一个个体的期望值小于零时,则该个体不参与选择。

对比实验表明,采用期望值法的性能高于前两种方法的性能。

(4) 排序选择方法

所谓排序选择方法是指在计算每个个体的适应值后，根据适应值大小顺序对群体中个体排序，然后把事先设计好的概率表按序分配给个体，作为各自的选择概率。所有个体按适应值大小排序，选择概率和适应值无直接关系而仅与序号有关。这种方法的不足之处在于选择概率和序号的关系必须事先确定。此外，它和适应值比例法一样都是一种基于概率的选择。

(5) 比例排序法

将比例法和排序法结合起来的比例排序法，即当群体中某个染色体的适应值远远大于其他染色体的适应值或群体中每个染色体的适应值相似时，按排序法进行后代选择，而在一般情形下采用比例法进行后代选择。这样既能利用两种方法各自的优点，又弥补了两种方法各自的缺点。

2. 交叉(crossover)算子

它又称重组(recombination)、配对(breeding)算子。

当许多染色体相同或者后代的染色体与上一代没有多大差别时，可通过染色体重组来产生新一代染色体。染色体重组是分两步骤进行的，首先在新复制的群体中随机选取两个个体，然后，沿着这两个个体(字符串)随机地取一个位置，二者互换从该位置起的末尾部分。如，有两个用二进制编码的个体 A 和 B。长度 $L=5$，$A=a_1a_2a_3a_4a_5$，$B=b_1b_2b_3b_4b_5$ 随机选择一整数 $k\in[1,L-1]$，设 $k=4$，经交叉后变为：

$$A = a_1a_2a_3 \mid a_4a_5,\quad A' = a_1a_2a_3b_4b_5$$
$$B = b_1b_2b_3 \mid b_4b_5,\quad B' = b_1b_2b_3a_4a_5$$

遗传算法的有效性主要来自选择和交叉操作，尤其是交叉在遗传算法中起着核心作用。目前有如下几种基本交叉方法。

(1) 一点交叉

一点交叉又叫简单交叉。具体操作是：在个体串中随机设定一个交叉点。实行交叉时，该点前或后的两个个体的部分结构进行互换，并生成两个新个体。

(2) 二点交叉

二点交叉的操作与一点交叉类似，只是设置两个交叉点(依然是随机设定)。一个二点交叉的例子表示如下：

个体 A　10⋮110⋮11 ⟶ 1001011 新个体 A'
配对个体　个体 B　00⋮010⋮00 ⟶ 0011000 新个体 B'
交叉点 1　交叉点 2

由此可见，2 个交叉点分别设定在第二个基因位和第三个基因位之间以及第五个基因位和第六个基因位之间。A、B 两个体在这两个交叉点之间的码串相互交换，分别生成新个体 A' 和 B'。对于二点交叉而言，若染色体长为 n，则可能有 $(n-2)(n-3)$ 种交叉点的设置。

(3) 多点交叉

多点交叉是前述两种交叉的推广，有时又被称为广义交叉(generalized crossover)。

一般来讲，多点交叉较少采用，因为它影响遗传算法的性能。即多点交叉不能有效地保存重要的模式。

(4) 一致交叉

所谓一致交叉是指通过设定屏蔽字(mask)来决定新个体的基因继承两个旧个体中哪个个体的对应基因。一致交叉的操作过程表示如下：当屏蔽字位为0时，新个体A'继承旧个体A中对应的基因，当屏蔽字位为1时，新个体A'继承旧个体B中对应的基因，由此生成一个完整的新个体A'。反之，可生成新个体B'。显然，一致交叉包括在多点交叉范围内。一个一致交叉的例子表示如下：

旧个体 A	001111
旧个体 B	111100
屏蔽字	010101
新个体 A'	011110
新个体 B'	101101

3. 变异(mutation)算子

选择和交叉算子基本上完成了遗传算法的大部分搜索功能，而变异则增加了遗传算法找到接近最优解的能力。变异就是以很小的概率、随机地改变字符串某个位置上的值。变异操作是按位(bit)进行的，即把某一位的内容进行变异。在二进制编码中，就是将某位0变成1，1变成0。变异发生的概率即变异概率P_m都取得很小(一般在0.001～0.02之间)，它本身是一种随机搜索，然而与选择、交叉算子结合在一起，就能避免由于复制和交叉算子而引起的某些信息的永久性丢失，保证了遗传算法的有效性。

遗传算法引入变异的目的有两个：一是使遗传算法具有局部的随机搜索能力。当遗传算法通过交叉算子已接近最优解邻域时，利用变异算子的这种局部随机搜索能力可以加速向最优解收敛。显然，此种情况下的变异概率应取较小值，否则接近最优解的模式会因变异而遭到破坏。二是使遗传算法可维持群体多样性，以防止出现未成熟收敛现象。此时变异概率应取较大值。

(1) 基本变异算子

基本变异算子是指对群体中的个体码串随机挑选一个或多个基因位并对这些基因位的基因值作变动(以变异概率P_m做变动)。{0,1}二值码串中的基本变异操作如下：

个体A 1 0 1 1 0 1 1 —变异→ 1 1 1 0 0 1 1 个体A'

变异基因位

(2) 逆转算子

逆转算子是变异算子的一种特殊形式。它的基本操作内容是：在个体码串中随机挑选两个逆转点，然后将两个逆转点间的基因值以逆转概率P_i逆向排序。{0,1}二值码串的逆

转操作如下：

$$\text{个体}A\quad 10\ \underline{11010}00 \xrightarrow{\text{逆转}} 10\ 01011\ 00\quad\text{个体}A'$$

逆转点

由此可见，通过逆转操作，个体中从基因位 3 至基因位 7 之间的基因排列得到逆转，即从 11010 序列变成了 01011 序列。这一逆转操作可以等效为一种变异操作，但是逆转操作的真正目的并不在变异（否则仅用变异操作就行了）而在实现一种重新排序操作。所谓重新排序是指对个体中基因排列进行重新组合，但并不影响该个体的特征。在自然界生物的基因重组中就有这种重新排序的机制。对遗传算法而言，采用这种重新排序，目的是为了提高积木块（高适应度个体）的繁殖率。实际上，在用遗传算法求解某些问题时，群体中的有些个体的基因排序常常会出现这样的情况，即对形成积木块有用的某些基因分离较远，此时采用一般的交叉会破坏相应的积木块的生成。因此，有必要对这些基因进行重新排序但又不损坏整个个体的特征（即适应值）。

（3）自适应变异算子

该算子与基本变异算子的操作内容类似，惟一不同的是变异概率 P_m 不是固定不变，而是随群体中个体的多样性程度而自适应调整。一般是根据交叉所得两个新个体的海明距离进行变化。海明距离越小，P_m 越大，反之 P_m 越小。

遗传算法中，交叉算子因其全局搜索能力而作为主要算子，变异算子因其局部搜索能力而作为辅助算子。遗传算法通过交叉和变异这一对相互配合又相互竞争的操作而使其具备兼顾全局和局部的均衡搜索能力。所谓相互配合，是指当群体在进化中陷于搜索空间中某个超平面而仅靠交叉不能摆脱时，通过变异操作可有助于这种摆脱。所谓相互竞争，是指当通过交叉已形成所期望的模式时，变异操作有可能破坏这些模式。因此，如何有效地配合使用交叉和变异操作，是目前一个重要的研究内容。

10.4.3 遗传算法简例

问题：求解 $f(x)=x^2$ 在[0,31]上的最大值。

1. 初始种群

（1）编码：用 5 位二进制表示 x，有

$$x=0\to 0\ 0\ 0\ 0\ 0,\quad x=31\to 1\ 1\ 1\ 1\ 1$$

（2）初始种群

随机产生 4 个个体：13 ，24 ，8 ，19（分别用二进制表示）。

（3）适应值 f_i

直接用目标函数作为适应值：$f(x)=x^2$

① 非负；② 逐步增大。

（4）选择率 P_i 和期望值

选择率：
$$P_i = f_i / \sum_i f_i$$

平均适应值：　　　　　　$\underline{f} = \sum_i f_i / n$

期望值：　　　　　　　　$f_i / \underline{f}$

(5) 实选值

期望值取整数，具体计算如表 10.1 所示。

表 10.1　初始种群参数计算

编　号	初始种群位串	参数值 x 值	目标适应值 $f(x)=x^2$	选择率 $f_i/\sum_i f_i$	期望值 $f_i/\underline{f}$	实选值
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1
总和 $\sum$			1170	1.00	4.00	4.0
平均值			293	0.25	1.00	1.0
最大值			576	0.49	1.97	2.0

2. 遗传一代

具体计算如表 10.2 所示。

表 10.2　初始种群遗传过程

选择后的交配池（下划线部分交叉）	交叉对象（随机选择）	交叉位置（随机选择）	新的种群	x	$f(x)=x^2$
0 1 1 0 $\underline{1}$	2	4	0 1 1 0 0	12	144
1 1 0 0 $\underline{0}$	1	4	1 1 0 0 1	25	625
1 1 $\underline{0\ 0\ 0}$	4	2	1 1 0 1 1	27	729
1 0 $\underline{0\ 1\ 1}$	3	2	1 0 0 0 0	16	256
总和 $\sum$					1754
平均值					439
最大值					729

表 10.1、表 10.2 的具体说明如下：

(1) 选择(繁殖)

在种群中，实选值(期望值)高者多繁殖；实选值(期望值)低者少繁殖或不繁殖。繁殖(复制)的个体放入交配池中。

(2) 交叉

随机选择交配对象(相同个体不交配)，如个体 1 和 2，3 和 4。随机选择交叉点进行交叉。

(3) 变异

取变异概率 $P_e=0.01$，表示每 100 个个体中有 1 个个体的 1 位发生变异。上例中未进行个体变异。

遗传得到的新的种群，其平均值和最大值都有很大提高。

均值：293 → 439。

最大值：576 → 729。

新种群中 4 个个体，有 2 个变好：25，27；2 个变坏：12，16。

3. 遗传第二代

新种群的参数计算如表 10.3 所示，新种群的遗传过程如表 10.4 所示。

表 10.3　新种群参数计算

编　号	初始种群位串	参数值 x 值	目标适应值 $f(x)=x^2$	选择率 $f_i/\sum_i f_i$	期望值 $f_i/\bar{f}$	实选值
1	0 1 1 0 0	12	144	0.08	0.33	0
2	1 1 0 0 1	25	625	0.36	1.42	1
3	1 1 0 1 1	27	729	0.42	1.66	2
4	1 0 0 0 0	16	256	0.15	0.58	1
总和 $\sum$			1754	1.00	4.00	4.0
平均值			439	0.25	1.00	1.0
最大值			729	0.42	1.66	2.0

表 10.4　新种群的遗传过程

选择后的交配池（下划线部分交叉）	交叉对象（随机选择）	交叉位置（随机选择）	新的种群	x	$f(x)=x^2$
1 1 0 0 1	2	1	1 1 0 1 1	27	729
1 1 0 1 1	1	1	1 1 0 0 1	25	625
1 1 0 1 1	4	3	1 1 0 0 0	24	576
1 0 0 0 0	3	3	1 0 0 1 1	19	361
总和 $\sum$					2291
平均值					572
最大值					729

单纯用交叉而没有用变异，则遗传多少代得不到最优解 31(11111)。主要是第三位所有个体都是 0，这样只能得到 27(11011)次优解。

若在第四位中挑选一个个体进行变异，由 0 变成 1，再进行遗传将会得到最优解。

10.4.4　遗传算法的特点

遗传算法是模拟自然选择和生物遗传机制的优化算法，利用 3 个遗传算子产生后代，通过群体的迭代，使个体的适应性不断提高，最终群体中适应值最高的个体即是优化问题的最优或次优解。遗传算法与传统的优化方法有不同的特点。

1. 遗传算法是进行群体的搜索

传统的优化方法是从一个点开始搜索。如爬山法(climbing)是从当前点邻近的点中选

出新点,如果新点的目标函数值更好,那么该新点就变成当前点,否则就选择和测试其他邻近点。如果目标函数值没有更进一步的改进,则算法终止。很显然,爬山法只能提供局部最优解,它依赖于初始点的选择。

遗传算法是对多个个体进行群体的搜索,即在问题空间中不同区域进行搜索,构成一个不断进化的群体序列。对于复杂问题的多峰情况,遗传算法也能以很大的概率找到全局最优解。

2. 遗传算法是一种随机搜索方法

遗传算法使用 3 个遗传算子,选择算子通过选择概率复制个体。交叉算子通过交叉概率在交配池中决定配对的个体是否需要进行交叉操作。变异算子通过变异概率确定某些基因位上的值进行变异。可见,3 个遗传算子都是随机操作,利用概率转移规则产生好的后代,引导其搜索过程朝着更优化的解空间移动。可见遗传算法虽然是一个随机搜索方法,但是它是高效有方向的搜索,而不是一般随机搜索方法那种无方向的搜索。

3. 遗传算法处理的对象是个体,而不是参变量自身

遗传算法要求将优化问题的参变量编码成长度有限的位串个体,即参变量是个体的组成部分。通过遗传算子操作位串个体,并从中找出高适应值的位串个体。遗传算法不是对参数变量进行直接操作。

编码操作可直接对结构对象进行操作。结构对象泛指集合、序列、矩阵、树、图、链和表等一维或二维结构形式的对象。这一特点使得遗传算法具有广泛的应用领域。

4. 遗传算法不需要导数或其他辅助信息

一般传统的搜索算法需要一些辅助信息,如梯度算法需要求导数,当这些信息不存在时(如函数不连续时),这些算法就失效。而遗传算法只需要适应值信息,用它来评估个体,引导搜索过程朝着搜索空间的更优化的解区域移动。

5. 隐含并行性

遗传算法实质上是模式的运算。对于一个长度为 l 的串,其中隐含着 2^l 个模式。若群体规模为 n,则其中隐含的模式个数介于 2^l 和 $n2^l$ 之间。Holland 指出,遗传算法实际上是对 n 个位串个体进行运算,但却隐含地处理了大量的模式,这一性质称为隐含并行性(implicit parallelism)。

隐含的并行性是遗传算法优于传统的搜索方法的关键所在。

10.5 基于遗传算法的分类学习系统

10.5.1 概述

1978 年 Holland 等人实现了第一个基于遗传算法的机器学习系统 CS-1。该系统由消息表(message list)、分类器(classifier)的字符串规则、遗传算法及一个信息分配机制组成。

他还提出了桶队(bucket brigade)算法。1980 年 Smith 实现了分类器系统 LS-1。尽管 LS-1 诞生在 CS-1 之后,但 LS-1 系统在若干重要的方面与 CS-1 有根本性的差别。具体表现在字符串规则、染色体表示方法、搜索结构的形成以及遗传操作算子的应用上。LS-1 系统影响更大。

分类器系统是一种学习字符串规则(又称分类器)的学习系统,它由规则与消息(rule and message)系统、信任分配(apportionment of credit)系统及遗传算法 3 个主要部分组成,其中规则与消息系统是产生式系统的一种特殊形式。产生式规则的一般形式为 IF＜condition＞THEN＜action＞。它具有计算完备性,且其描述也较方便,一条规则或一个规则集往往能将一种复杂的情况非常紧凑地描述出来。因而它为众多的专家系统所采用。在分类器系统中,对产生式规则的语法做了很大的限制,采用了定长的表示形式,从而适于采用遗传操作。

传统的专家系统在每一次匹配中采用单条规则激活的串行运行方式。分类器系统采用了并行激活方式,即在每一匹配周期,允许多条规则被同时激活,只有在出现两个互斥的动作或当匹配的规则集大小超出消息表的容量时,才考虑规则的选择问题。

传统的专家系统中的规则和规则相应的重要程度(strength)是事先由程序设计者根据专家经验给出,是固定不变的。而分类器系统是一个自适应的学习系统,使用的是概率转换规则,而不是确定性规则。其规则和相应的重要程度是不固定的,这是需要学习的关键信息。

10.5.2 遗传分类学习系统 GCLS 的基本原理

我们研制了一种新的遗传分类器学习系统(genetic classifier learning system,GCLS),与基本的分类器系统相比,GCLS 系统采用了训练和测试同时进行的策略,使得系统能够在训练后继续学习,从而能更好地适应不断变化的客观环境。GCLS 系统还设计了工作和精练两种不同的分类器,通过精练分类器对规则的进一步处理,减少了所获规则的冗余性。GCLS 系统中设计的信任分配机制可有效地处理训练样本带有噪声和异常特例等问题,同时体现了规则与训练样本的统计规律,使得判别结果容易用背景知识进行定性、定量相结合的解释,从而可获得与客观环境相容的判别规则。

1. GCLS 系统结构

遗传分类学习系统 GCLS 的结构如图 10.15 所示。

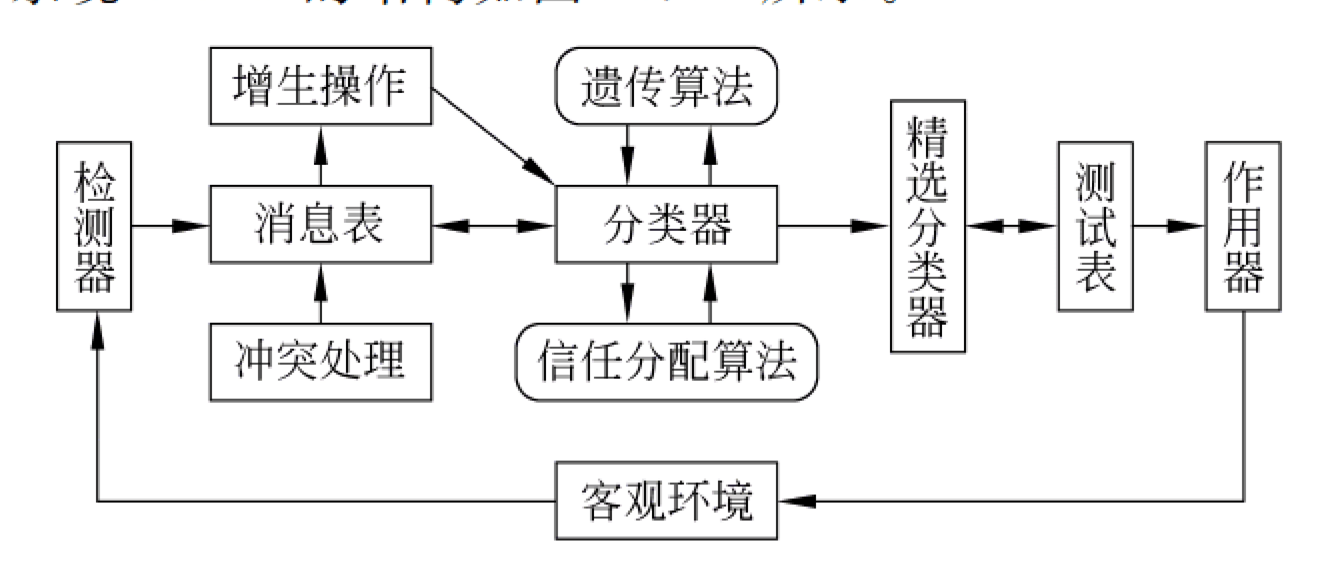


图 10.15　遗传分类学习系统 GCLS 的结构

客观环境信息通过分类器系统的检测器(detector)被编码成有限长的消息(messages)。然后发往消息表;消息表中的消息触发位串规则(称为分类器),被触发的分类器又向消息表发消息,这些消息又有可能触发其他的分类器或引发一个行动,通过作用器(effector)作用于客观环境。

(1) 检测器(detector)

将环境信息由条件部分和结论部分组成的训练的例子集,编码成二进制字符串的消息。一条消息 M_i 是一个二元组,其形式为 $M_i=[x_i,y_i]$,其中:i 为消息号;x 为条件部分,即训练例子的各特征编码,$x_i\in\{0,1\}^n$;y 为结论部分,即训练例子的类别,$y_i\in\{0,1\}^m$。例如:[(10001011),(1011)]是一条由一个 8 位条件和 4 位结论组成的消息。

(2) 消息表(message list)

消息表包含当前所有的消息(训练例子集)。每个消息由 message 和 action 两部分组成。

(3) 分类器(classifier)

分类器系统与一般的机器学习系统不同,最后所获得的规则中包含通配符#,会出现大量的冗余规则,如:1##0,1110 是一致的。一般来说,应该使系统产生最小的规则集,获得较高的性能。规则集越小,系统的时间性能当然越好。

一个分类器是由当前遗传产生的一条规则组成,分类器表由所有分类器组成,构成了规则集。一个规则 C_i 是一个三元组,形式如下:

$$C_i=[U_i,V_i,\text{fitness}_i]$$

其中,U_i 是条件部分(condition),$U_i\in\{0,1,\#\}^n$,#表示通配符;V_i 是结论部分(action),$V_i\in\{0,1\}^m$;fitness_i 是规则 i 的适应值,又是一个二元组,其形式如下:

$$\text{fitness}_i=[\text{fit1},\text{fit2}]$$

其中:fit1、fit2 均为正整数,分别表示在该规则覆盖的范围内,与规则结论一致和不一致的消息个数。

在分类器中,将最后获得的规则放入精练分类器中。

(4) 测试表(test list)

测试表是由所有测试例子组成,一个测试例子 T_i 也是一个同消息形式一样的二元组,只是它的结论部分 $y_i\in\{*\}^m$,*表示未确定。当它到精练分类器匹配规则后,其结论部分 y_i 就被赋值成与消息 M_i 完全一样形式,即 $y_i\in\{0,1\}^m$,变成一条新的消息。结论可直接作用于环境,也可通过环境将新消息反馈给系统,以便系统能继续学习下去,从而更好地适应不断变化的客观环境。

(5) 作用器(effector)

作用器将所有测试例子的判别结果(类别)转换成具体问题的输出值,并作用于环境。

遗传分类学习系统 GCLS 规则生成流程,如图 10.16 所示。

2. GCLS 系统的主要算法

(1) 信任分配算法(credit assignment algorithm,CAA)

信任分配实质上是对各条规则(分类器)作用于环境的有效性进行评价,而本系统中的

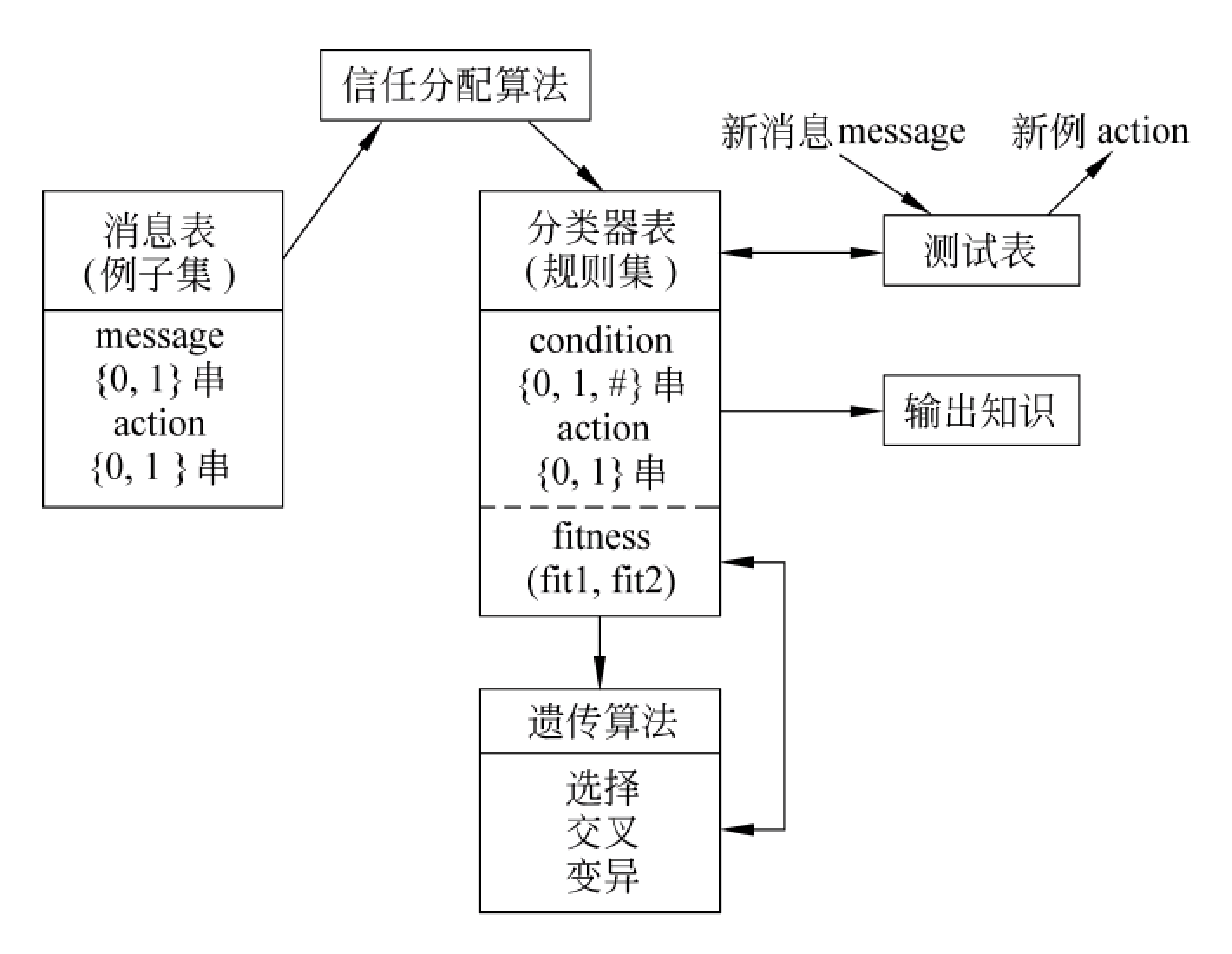


图 10.16　GCLS 规则生成过程

环境就是前面所说的训练例子集，将规则（分类器）与消息表中的消息逐个匹配，根据匹配的成功与否，来修改规则的适应值，以保证好的规则的生存，不适应的规则的消亡，其主要步骤如下：

① 初始化规则的适应值，即：fit1←0，fit2←0。

② 从消息表[M]中取出一条消息，与工作分类器[WC]中的规则逐个进行比较。

IF　条件和结论均匹配即消息的 message 与规则的 condition 匹配，消息的 action 与规则的 action 匹配，THEN　fit1←fit1＋1；

IF　条件匹配，结论不匹配，THEN　fit2←fit2＋1；

IF　条件不匹配，THEN　fitness←fitness。

③ 返回步骤②，直到[M]中的消息全部取完。

(2) 遗传算法(genetic algorithms)

遗传算法是用来产生新的规则。在 GCLS 系统中，遗传算法的调用是在工作分类器中每一新的种群产生之后。系统采用了一种限制交配策略，也就是本地算子中的受限交配，即只允许同类（规则的结论部分相同）的规则进行交叉。这样，对同一结论的规则，只允许其条件部分进化。假如规则的条件和结论同时进化，就可能引起种群不收敛的情况。此外，产生的新规则并不取代老规则，而是与老规则合并到一起，形成工作分类器的新的初始种群。

GCLS 中遗传算法的主要步骤如下：

① 在工作分类器中，根据与各规则适应值成正比的概率，选择复制出 k 个规则。选择复制策略因具体问题而不同。本系统中采用了比例法选择复制。按 $f_i \Big/ \sum_j f_j$ 取整（f_i 是 X_i 的适应值；$\sum_j f_j$ 是种群中各规则的适应值之和），来决定第 i 个规则 X_i 在下一代中应复制其自身的数目 k_i，而 $K = \sum_i k_i$。

② 采用遗传算子(交叉、变异),重新产生 k 个新的规则。在 GCLS 中,按一定的概率 P_c 从①中随机地选择出一对规则进行交叉,同样,也是按一定的概率 P_m 对规则中的某些位进行变异。这里的交叉概率 P_c 和变异概率 P_m 都是经验参数,在不同应用问题中的取值都是不同的。

(3) 合并操作(merge operation)

采用合并操作旨在减少冗余规则。

① 对于工作分类器[WC]中初始种群的每一规则,若其对应的 fit1 恒不等于 0,且 fit2 等于 0,则保留,否则淘汰。

② 将保留下来的规则两两匹配。设 R1、R2 为两个保留下来的规则。

IF R1⊇ R2,且 fit1(R1)=fit1(R2), THEN 保留 R2,淘汰 R1

IF R1⊇ R2,且 fit1(R1) > fit1(R2), THEN 保留 R1,淘汰 R2

(4) 冲突处理(conflict process)

一般的分类器系统不包括矛盾例子的处理,而在实际应用领域尤其在预测领域,这种情况经常出现,如天气预报。所以系统要对这些矛盾例子能够处理。GCLS 系统中设计的冲突处理是将消息表[M]中的消息两两匹配,对于那种只有条件匹配,而结论不匹配的消息作为冲突消息,记录下来,并都从[M]中删除。在分类器中删除已生成的冲突规则。

(5) 增生操作(supplement operation)

如果分类器[C]中没有一个与消息匹配的规则,则用增生操作生成一个与之相匹配的规则。在消息位串上对条件部分的每一位按系统给定的♯的生成率进行变异。若发生变异则由 1 或 0 改为♯,否则不变。然后将变异过的消息作为新的规则的条件部分,结论部分随机产生。新生成的规则加入到[C]中的方法有两种:一是用新生成的规则置换掉[C]中的适应值最小的规则。二是直接加入到[C]中,只有当[C]的增长超过一定限度时才进行淘汰。这样做的好处是在系统运行的初期,当适应值的强弱差别还不明显时,能较好地避免将有发展潜力、好的规则淘汰掉。在本系统中,采用了后一种方法。

此外,在 GCLS 系统中采用了训练与测试同时进行。一般的分类器系统同现存的机器学习系统一样:训练与测试是分开进行的,规则的获取完全依赖于训练例子的选取的好坏。例如,训练例子中正反例的比例应与实际问题中正反例的比例相同,这一般是不可能做到的,且选取的训练例子不可能包含实际问题中的所有情况。而 GCLS 的这种策略使系统能在训练后继续学习,就能保证不依赖于选取的例子,从而能更好地适应不断变化的客观环境,得到更符合实际的规则。

3. GCLS 系统获取规则的过程

GCLS 系统的学习过程是一个获取规则的过程。规则的获取是通过初始化一个随机的种群(分类器),而后触发系统的信任分配机制和遗传算法等操作,直到获得一组源于环境信息(训练集)的、达到期望状态或特征的规则(分类器),再把最后获得的规则复制到一个精练文件(精练分类器)中,以供下一步测试未知例子的类别使用,至此,GCLS 系统的一个学习过程就已结束。

在 GCLS 系统中一次学习过程的结束是当目前分类器已收敛,即种群的规则与其父代

完全相同,并且各规则的适应值已连续 p 次保持不变,也就是说,当前工作种群已不再进化了,p 是系统根据不同的应用问题而事先设置的一个参数,在本系统应用实例中 p 均取 100。

GCLS 系统的执行步骤可概括如下：

(1) 初始化 GCLS 的所有预置参数。如每一分类器[C]中初始规则数目 n;交叉、变异概率 P_c、P_m;判断分类器收敛的参数 p 等;初始化分类器[C],设为初始种群 0,随机产生 n 个规则,并给每个规则赋一个相等的初始适应值。

(2) 将环境信息(训练集)通过检测器编码成二进制消息放入消息表[M]中。

(3) 对[M] 进行冲突处理。将[M]中的消息进行两两匹配,把只有条件匹配而结论不匹配的消息作冲突处理后,直接送往精练分类器[RC]中。

(4) 对初始种群 0 调用信任分配算法,修改其中的规则适应值。如果种群 0 中没有与消息匹配的规则,则进行增生操作,生成一个相匹配的规则,将该规则直接加入到种群 0 中。

(5) 对种群 0 进行合并操作,合并后的种群设为种群 1。

(6) 假如种群 1 已收敛,则复制该种群的规则到精练分类器[RC]中,转向步骤(9)。

(7) 调用遗传算法,生成新一代种群 2,将其与种群 1 合并,而后送给种群 0,从而形成新的种群 0。

(8) 返回步骤(4)。

(9) 对测试表[T]调用精练分类器规则,生成[T]的结论部分。

(10) 将[T]送往作用器,转换成实际的输出值,以便作用于环境。

10.5.3 遗传分类学习系统 GCLS 的应用

1. 应用说明

这是一个学习识别脑出血和脑血栓两种疾病的诊断规则的应用实例,这个问题实际上是从大量已知患者病例(训练例子集)中找到这两类病的识别规则。

在这一应用实例中,实际上只有两种类别:脑出血和脑血栓。

为了作出判断,应当考虑如下几个方面的特征(属性):

(1) 病人的既往史,包括：高血压(有：01,无：00);动脉硬化(有：01,无：00)。

(2) 起病方式(快：01,慢：00)。

(3) 局部症状,包括：

a. 偏瘫(是：01,否：00)。

b. 瞳孔不等大(是：01,否：00)。

c. 两便失禁(是：01,否：00)。

d. 语言障碍(是：01,否：00)。

e. 意识障碍(无：00,深度：01,轻度：10)。

(4) 病理反射(阳：01,阴：00)。

(5) 膝腱反射(无：00,活跃：01,不活跃：10)。

(6) 病情发展(快：01,慢：00)。

上面是从 6 个方面 12 个特征来识别诊断患者到底得的是脑出血还是脑血栓。

2. 获取知识

我们从 200 多个脑出血和脑血栓病人的病例中选出 30 个病例作为训练样本，选出 100 个作为测试样本。

本实例采用二进制编码方式。每个训练例子是由 12 个特征和 1 个类别组成，每个特征和类别都由 2 位二进制字符表示。那么，将例子编码成二进制字符串的消息就是一个由 24 位条件和 2 位结论组成的二元组，例如消息 M=[(010001010101010110100101),(01)]。

训练集是由 15 个脑出血和 15 个脑血栓患者组成 30 个训练样本。本实验在对 30 个训练样本进行学习后，得到 12 个规则，学习终止于第 170 代。

获取的主要规则如下：

(1) 高血压=有∧瞳孔不等大=是∧膝腱反射=不活跃 →脑出血 (11)
(2) 瞳孔不等大=是∧语言障碍=是 →脑出血 (12)
(3) 高血压=有∧起病方式=快∧意识障碍=深度 →脑出血 (13)
(4) 高血压=有∧病情发展=快 →脑出血 (15)
(5) 高血压=有∧动脉硬化=有∧起病方式=慢 →脑血栓(13)
(6) 动脉硬化=有∧病情发展=慢 →脑血栓 (15)
(7) 动脉硬化=有∧意识障碍=无 →脑血栓 (12)

以上括号内的数值表示该规则的适应值。

习　题

1. 说明神经网络的 MP 模型和 Hebb 规则原理。
2. 神经元网络的几何意义是什么？
3. 说明下列样本是什么类型样本，为什么？

(1)

输　入		输　出
x_1	x_2	d
0	0	0
0.5	0.5	1
1	1	0

(2)

输　入		输　出
x_1	x_2	d
0	0	0
0.5	0	1
1	1	0

4. BP 模型中误差公式：$\delta_i = f'(\text{net}_i)\sum_k \delta_k w_{ki}$ 的含义是什么？

5. 对如图 10.17 所示的 BP 神经网络，写出它的计算公式（含学习公式），并对其初始权值以及样本 $x_1=1, x_2=0, d=0$ 进行一次神经网络计算和学习（该系数 $\eta=1$，各点阈值为 0）。

作用函数简化为

$$y = f(x) = \begin{cases} 0.95, & x \geqslant 0.45 \\ x+0.5, & -0.45 < x < 0.45 \\ 0.05, & x \leqslant -0.45 \end{cases}$$

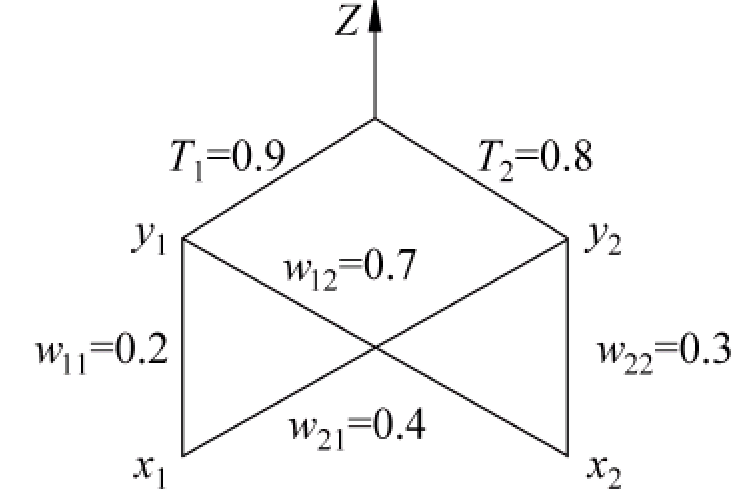


图 10.17　BP 神经网络图

6. 编制 BP 网络模型程序，完成异或问题的计算。

7. 遗传算法中的染色体与基因是如何表示的？

8. 遗传算法的处理流程是怎样的？

9. 遗传算法中如何确定适应值函数？

10. 选择算子有几种？各自的计算方法是什么？

11. 交叉算子有几种？各自的操作方法是什么？

12. 变异算子有几种？各自的操作方法是什么？

13. 遗传算法的特点有哪些？

14. 从遗传算法的简例中，说明 3 个遗传算子的作用是什么？

15. 遗传分类学习系统 GCLS 规则生成过程的示意图是什么？

16. GCLS 系统的信任分配算法的步骤是什么？

17. GCLS 系统的遗传算法的主要步骤是什么？

18. GCLS 系统用于脑出血、脑血栓疾病诊断的个体编码方式是什么？

第 11 章　文本挖掘与 Web 挖掘

11.1　文本挖掘概述

11.1.1　文本挖掘的基本概念

在现实世界中，我们面对的数据大都是文本数据，由各种数据源（如新闻文章、研究论文、书籍、数字图书馆、电子邮件和 Web 页面）的大量文本组成。由于文本的信息量的飞速增长，如电子出版物、电子邮件、CD-ROM 和 Web 等，文本数据的数量急剧增长。

文本数据是半结构化数据，既不是完全无结构的也不是完全结构化的。例如，文本可能包含结构字段，如标题、作者、出版日期、长度、分类等，也可能包含大量的非结构化的文本，如摘要和内容。

文本挖掘（text mining）也称为文本数据挖掘（text data mining）。文本挖掘一词出现于 1998 年第十届欧洲机器学习会议（the European Conference On Machine Learning，ECML'98）上，首次进行了关于文本挖掘的专题讨论会。组织者 Kodratoff 明确地定义文本挖掘的概念，并分清它与“信息检索”（information retrieval）的不同点和共同点。他认为文本挖掘的目的是从文本集合中搜寻知识，并不试图改进自然语言理解，并不要求对自然语言的理解达到多高的水平，而只是想利用该领域的成果，试图在一定的理解水平上尽可能多地提取知识。因此，文本挖掘需要数据挖掘、语言学、数据库以及文本标引和理解方面的专家的参与。

1．概念

文本挖掘是从大量文本数据中提取以前未知的、有用的、可理解的、可操作的知识的过程。文本数据包括技术报告、文本集、新闻、电子邮件、网页、用户手册等。文本挖掘对单个文本或文本集（如 Web 搜索中返回的结果集）进行分析，从中提取概念，并按照指定的方案组织、概括文本，发现文本集中重要的主题。它除了从文本中提取关键词外，还要提取事实、作者的意图、期望和主张等。这些知识对许多应用目标，如市场营销、趋势分析、需求处理等，都是很有用的。

相对于数据挖掘，文本挖掘面临的主要问题是挖掘的对象是半结构化或非结构化的，而且自然语言文本中包含多层次的歧义（如词汇、句法、语义、语用等）等。

2．主要任务

文本挖掘的主要任务是：

（1）短语提取，即在读取大量的非结构化文本时，应用自然语言处理技术提取文本集中

所有相关的短语。提取时要处理同义词和词义模糊现象。可以形象地把文本挖掘看作是一支荧光笔,它通读文本时高亮度显示有关的短语,这些短语放在一起就可以得到对文本的一个较好的理解。

(2) 概念提取(聚类),即对短语之间的关系建立一个"词汇网";将相关的短语分组,并增强这些组中最重要的特征;最后得到的模式反映了该文本集中的主要概念;然后,从提取出的概念集中发现未知的知识。

(3) 可视化显示和导航,即对挖掘得来的信息(如词频、相关频率、时事性话题、地域依赖信息、时间序列等)可以从多个视角进行分析。

3. 文本挖掘与数据挖掘

文本挖掘与数据挖掘相比,它们的相似点在于两者都处理大量的数据,都可归属到知识发现领域中。它们之间的差别在于许多经典的数据挖掘算法,如数值预测、决策树等都不太适用于文本挖掘,因为它们依赖于结构化的数据。而短语或概念关联分析等工作则是文本挖掘所独有的,如表 11.1 所示。

表 11.1　文本挖掘与数据挖掘的区别

项　目	数据挖掘	文本挖掘
研究对象	用数字表示的、结构化的数据	无结构或者半结构化的文本
对象结构	关系数据库	自由开放的文本
目标	获取知识,预测以后的状态	提取概念和知识
方法	归纳学习、决策树、神经网络、粗糙集、遗传算法等	提取短语、形成概念、关联分析、聚类、分类
成熟度	从 1994 年开始得到广泛应用	从 2000 年开始得到广泛应用

11.1.2　文本特征的表示

与数据库中的结构化数据相比,文本具有有限的结构,或者根本就没有结构,即使具有一些结构,也是着重于格式,而非文本内容。不同类型文本的结构也不一致。此外,文本的内容是人类所使用的自然语言,计算机很难处理其语义。文本信息源的这些特殊性使得现有的数据挖掘技术无法直接应用于其上。所以需要对文本进行预处理,抽取代表其特征的元数据。这些特征可以用结构化的形式保存,作为文本的中间表示形式。

文本特征指的是关于文本的元数据,分为两种:描述性特征,例如文本的名称、日期、大小、类型等;语义性特征,例如文本的作者、机构、标题、内容等。描述性特征易于获得,而语义性特征则较难得到。对于内容这个难以表示的特征,首先要找到一种能够被计算机所处理的表示方法。

矢量空间模型(VSM)是近年来应用较多且效果较好的表示文本特征的方法。在该模型中,文本空间被看作是由一组正交词条矢量所形成的矢量空间,每个文本 d 表示为其中的一个规范化特征矢量:

$$V(d) = (t_1, w_1(d); \cdots; t_i, w_i(d); \cdots; t_n, w_n(d))$$

其中 t_i 为词条项,$w_i(d)$ 为 t_i 在 d 中的权值。可以将 d 中出现的所有单词作为 t_i,也可以要

求 t_i是 d 中出现的所有短语，从而提高内容特征表示的准确性。$w_i(d)$一般被定义为 t_i在 d 中出现频率 $tf_i(d)$的函数，即 $w_i(d)=\Psi(tf_i(d))$。常用的 Ψ 有：

1. 布尔函数

$$\Psi = \begin{cases} 1, & tf_i(d) > 0 \\ 0, & tf_i(d) = 0 \end{cases}$$

2. 平方根函数

$$\Psi = \sqrt{tf_i(d)}$$

3. 对数函数

$$\Psi = \log(tf_i(d) + 1)$$

4. TFIDF 函数

$$\Psi = tf_i(d) \times \log\left(\frac{N}{n_i}\right)$$

其中，N 为所有文本的数目，n_i为含有词条 t_i的文本数目。

11.1.3 文本特征的提取

特征提取主要是识别文本中代表其特征的词项。提取过程是自动的，提取的特征大部分是文本集中表示的概念。文本特征分为一般特征和数字特征，其中一般特征主要包括动词和名词短语，如人名、组织名等；数字特征主要包括日期、时间、货币以及单纯数字信息。这些特征包含重要的信息，因此特征提取是一种强有力的文本挖掘技术。通过文本特征抽取，用于记录文本的特征，可以更好地组织文本，如文本的存储、检索、过滤、分类和摘要等。

中文姓名识别属于中文信息处理中未登录词处理的范畴。中文姓名在文章中的出现频率虽然不高，但绝非可以忽略，因为中文姓名本身包含着重要的信息，它可能是整个句子甚至整个段落的语义中心，如果不予处理，将影响文本挖掘的性能。数字特征反映一定的信息，但不能表达文本的中心思想，通常只作文本挖掘中的参考信息。姓名特征提取算法所提取的姓名特征可作为文本内容的特征表示。

构成文本的词汇，数量是相当大的，因此，表示文本的向量空间的维数也相当大，可以达到几万维，因此需要压缩维数，这样做的目的主要有两个：第一，为了提高程序的效率，提高运行速度；第二，所有几万个词汇对文本分类的意义是不同的，一些通用的、各个类别都普遍存在的词汇对分类的贡献小，在某特定类中出现比重大而在其他类中出现比重小的词汇对文本分类的贡献大。

为了提高分类精度，对于每一类，应去除那些表现力不强的词汇，筛选出针对该类的特征项集合。目前存在多种筛选特征项的算法，如根据词和类别的互信息量判断、根据词熵判断等。

例如，根据词和类别的互信息量进行特征项（能体现类别的词）抽取的判断算法过程如下：

（1）初始情况下，该特征项集合包含所有该类中出现的词。

（2）对于每个词，计算词 W_i和类别 C_j的互信息量 $I(W,C)$。

（3）对于该类中所有的词，依据上面计算的互信息量排序。

(4) 抽取一定数量的词(互信息量大的词)作为特征项,具体需要抽取多少特征项,目前无很好的解决方法,一般采用先定初始值,然后根据实验测试和统计结果确定最佳值,一般初始值定在几千左右。

(5) 将每类中所有的训练文本,根据抽取的特征项进行向量压缩,精简向量表示。

11.2 文本挖掘

11.2.1 文本挖掘功能层次

文本挖掘的功能可以用一个层次结构表示,如图 11.1 所示。

文本挖掘功能从顶端到底端说明如下。

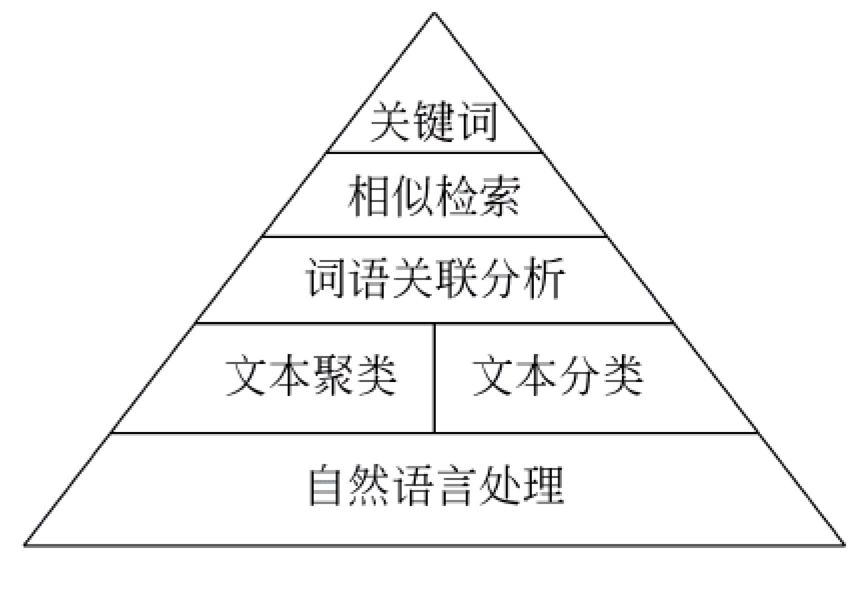


图 11.1 文本挖掘功能层次

1. 关键词检索

关键词建立倒排文件索引。简单的搜索引擎通常基于关键词检索相关文档,该技术与传统的信息检索使用的技术类似。

2. 相似检索

它与信息检索方法中的相似性检索方法类似,目的是找到相似内容的文本。

3. 词语关联分析

它不仅将注意力放在孤立词语的相同或相似信息上,而且聚焦在词语(包括关键词)之间的关联信息分析上。从而避免传统的信息检索技术带来的信息不精确和信息量过大等问题。

4. 文本聚类和文本分类

利用类似于数据挖掘的聚类和分类技术实现文本的聚类和分类。将文本在一个更高层次上进行抽象和整理。

5. 自然语言处理

这是最复杂的功能,它希望揭示自然语言处理技术的语义,进行文本语义挖掘。

目前文本挖掘主要是词语关联分析、文本聚类和文本分类工作。

11.2.2 关联分析

基于关键词或短语的关联分析首先收集经常一起出现的关键词或短语,然后找出其关联或相互关系。

关联分析首先要对文本数据进行词根处理,去除非用词等预处理,然后调用关联挖掘算法。在文本数据库中,每一文本被视为一个事务,文本中的关键词组可视为事务中的一组事

务项。文本数据库可表示为

{文本编号， 关键词集}

这样，文本数据库中关键词关联挖掘的问题就变成事务数据库中事务项的关联挖掘。

注意一组经常连续出现或紧密相关的关键词可形成一个词或词组。关联挖掘有助于找出复合关联(compound association)，即领域相关的词或词组，如“科技大学，大学”或“总统，克林顿”，或非复合关联，如“美元，参股，交易，总额，佣金，赌注，证券”。基于这些关联的挖掘称为“词级(term level)关联挖掘”(相对应的是字级的挖掘)。

词的识别和词组关联挖掘在文本分析中有两个优点：词和词组被自动标记，无需人去标记文本；挖掘算法的执行时间和无意义的结果将极大减少。

利用这种词和词组的识别，关联分析挖掘可以用于找出词或关键词间的关联。一些用户可能喜欢从给定关键词或词组中找出关键词或词组之间的关联，而有些用户可能希望找出一起出现的最大词集。因此，根据用户挖掘的需要，可以使用关联挖掘或最大模式挖掘算法。

11.2.3 文本聚类

文本聚类是一种典型的无教师的机器学习问题。目前的文本聚类方法大致可以分为层次聚类法和平面划分法两种类型。

1. 层次聚类法

对于给定的文本集合 $D=\{d_1,\cdots,d_i,\cdots,d_n\}$，层次聚类法的具体过程如下：

(1) 将 D 中的每个文本 d_i 看作是一个具有单成员的类 $c_i=\{d_i\}$，这些类构成了 D 的一个聚类 $C=\{c_1,\cdots,c_i,\cdots,c_n\}$；

(2) 计算 C 中每对类 (c_i,c_j) 之间的相似度 $\mathrm{sim}(c_i,c_j)$；

(3) 选取具有最大相似度的类对 $\max\limits_{c_i,c_j\in C}\mathrm{sim}(c_i,c_j)$，并将 c_i 和 c_j 合并为一个新的类 $c_k=c_i\cup c_j$，从而构成了 D 的一个新的聚类 $C=\{c_1,c_2,\cdots,c_{n-1}\}$；

(4) 重复上述步骤，直至 C 中剩下一个类为止。

该过程构造出一棵生成树，其中包含了类的层次信息，以及所有类内和类间的相似度。层次聚类法是最为常用的聚类方法，它能够生成层次化的嵌套类，且准确度较高。但是，在每次合并时，需要全局地比较所有类之间的相似度，并选择出最佳的两个类，因此运行速度较慢，不适合于大量文本的集合。

2. 平面划分法

平面划分法与层次聚类法的区别在于，它将文本集合水平地分割为若干个类，而不是生成层次化的嵌套类。对于给定的文本集合 $D=\{d_1,\cdots,d_i,\cdots,d_n\}$，平面划分法的具体过程如下：

(1) 确定要生成的类的数目 k；

(2) 按照某种原则生成 k 个聚类中心作为聚类的种子 $S=\{s_1,\cdots,s_j,\cdots,s_k\}$；

(3) 对 D 中的每个文本 d_i，依次计算它与各个种子 s_j 的相似度 $\text{sim}(d_i,s_j)$；

(4) 选取具有最大相似度的种子 $\max\limits_{s_j\in S}\text{sim}(d_i,s_j)$，将 d_i 归入以 s_j 为聚类中心的类 c_j，从而得到 D 的一个聚类 $C=\{c_1,c_2,\cdots,c_k\}$；

(5) 重复步骤(2)、(3)、(4)若干次，以得到较为稳定的聚类结果。该方法的运行速度较快，但是必须事先确定 k 的取值，且种子选取的好坏对聚类结果有较大影响。

11.2.4 文本分类

文本分类是一种重要的文本挖掘工作，由于存在大量的联机文本，分类便于对文本的检索和分析。

如何进行自动文本分类？一般的做法如下：首先，把一组预先聚类过的文本作为训练集。然后对训练集进行分析，以便得出各类的分类模式。这种分类模式通常需要一定的测试过程，不断地细化，用这些导出的分类模式对其他联机文本加以分类。

这一处理过程与关系数据库的分类相似，但还是存在本质的区别。关系数据库是结构化的：每个元组定义为一组“属性，值”对。文本数据库则不是结构化的，它没有“属性，值”对的结构。与一组文本相关的关键词并不能用一组属性或维来刻画。因此，通常面对关系数据库的分类方法，如决策树分析，并不适用于对文本数据库的分类。

对文本分类的有效方法是基于关联的分类，它基于一组关联的、经常出现的文本模式对文本加以分类。基于关联的分类方法处理过程如下：

(1) 通过简单的信息检索技术和关联分析技术提出关键词和词组。

(2) 使用已经有的词类，或基于专家知识，或使用某些关键词分类方法，生成关键词和词组的概念层次，或类层次结构。

(3) 词关联挖掘方法用于发现关联词，它可以最大化区分一类文本与另一类文本。这导致了对每一类文本有一组关联规则。这些分类规则可以基于其出现频率加以排序，并用于对新的文本分类。

基于关联的文本分类方法已经证明是有效的。对 Web 文本分类可以利用 Web 页面的链接信息，帮助文本类的识别。

文本分类是一种典型的有教师的机器学习问题，一般分为训练和分类两个阶段，具体过程如下。

1. 训练阶段

(1) 定义类别集合 $C=\{c_1,\cdots,c_i,\cdots,c_m\}$，这些类别可以是层次式的，也可以是并列式的；

(2) 给出训练文本集合 $S=\{s_1,\cdots,s_j,\cdots,s_n\}$，每个训练文本 s_j 被标上所属的类别标识 c_i；

(3) 统计 S 中所有文本的特征矢量 $V(s_j)$，确定代表 C 中每个类别的特征矢量 $V(c_i)$。

2. 分类阶段

(1) 对于测试文本集合 $T=\{d_1,\cdots,d_k,\cdots,d_r\}$ 中的每个待分类文本 d_k，计算其特征矢

量 $V(d_k)$ 与每个 $V(c_i)$ 之间的相似度 $\mathrm{sim}(d_k, c_i)$；

(2) 选取相似度最大的一个类别 $\max_{c_i \in C} \mathrm{sim}(d_k, c_i)$ 作为 d_k 的类别。

有时也可以为 d_k 指定多个类别，只要 d_k 与这些类别之间的相似度超过某个预定的阈值。如果 d_k 与所有类别的相似度均低于阈值，那么通常将该文本放在一边，由用户来做最终决定。如果这种情况经常发生，则说明需要修改预定义类别，然后重新进行上述训练与分类过程。在计算 $\mathrm{sim}(d_k, c_i)$ 时，有多种方法可供选择。最简单的方法是仅考虑两个特征矢量中所包含的词条的重叠程度，即：

$$\mathrm{sim}(d_k, c_i) = \frac{n(d_k, c_i)}{n_0(d_k, c_i)}$$

其中，$n(d_k, c_i)$ 是 $V(d_k)$ 和 $V(c_i)$ 具有的相同词条数目，$n_0(d_k, c_i)$ 是 $V(d_k)$ 和 $V(c_i)$ 具有的所有词条数目。最常用的方法是考虑两个特征矢量之间的夹角余弦。

11.3 Web 挖掘

目前万维网(WWW)是一个巨大的、分布广泛的和全球性的信息服务中心，涉及新闻、广告、消费信息、金融管理、教育、政府、电子商务和许多其他信息服务。Web 还包含了丰富和动态的超链接信息，以及 Web 页面的访问和使用信息，这为数据挖掘提供了丰富的资源。从广义上讲，Web 信息也是一类特别的文本信息，因此文本挖掘的各种技术也适合于 Web 挖掘，但是由于 Web 信息自身的特点，文本挖掘和 Web 挖掘应该区别对待。

11.3.1 Web 挖掘概述

1. Web 信息特点

(1) Web 信息特别庞大

Web 的数据量目前以几百 TB 计算，而且仍然在迅速地增长。许多机构和社团都在把各自大量的可访问信息置于网上。

(2) Web 信息非常复杂

Web 可以看作一个巨大的数字图书馆。然而，这一图书馆中的大量文本并不根据任何有关排列次序加以组织。它没有分类索引，更没有按标题、作者、封面页、目录等的索引。对在这样一个图书馆中搜索希望得到的信息是极具挑战性的。

(3) Web 信息是动态的

Web 不仅以极快的速度增长，而且其信息还在不断地发生着更新。新闻、股票市场、公司广告和 Web 服务中心都在不断地更新着各自的页面。链接信息和访问记录也在频繁地更新之中。

(4) Web 信息使用者复杂

Web 面对的是一个广泛的形形色色的用户群体。目前因特网上连接有约 5000 万台工作站，其用户群仍在不断地扩展中。各个用户可以有不同的背景、兴趣和使用目的。

(5) Web 信息中的“垃圾”非常多

一个人只是关心 Web 上的很小一部分信息，Web 所包含的其余信息对用户来说是不感兴趣的，而且会淹没所希望得到的搜索结果。

2. Web 挖掘分类

可以将 Web 挖掘一般地定义为：从 WWW 的资源和行为中抽取感兴趣的、有用的模式和隐含的信息。一般地，Web 挖掘可分为 3 类：Web 内容挖掘(Web content mining)、Web 结构挖掘(Web structure mining)和 Web 应用挖掘(Web usage mining)。

图 11.2 给出了 Web 挖掘的分类图。

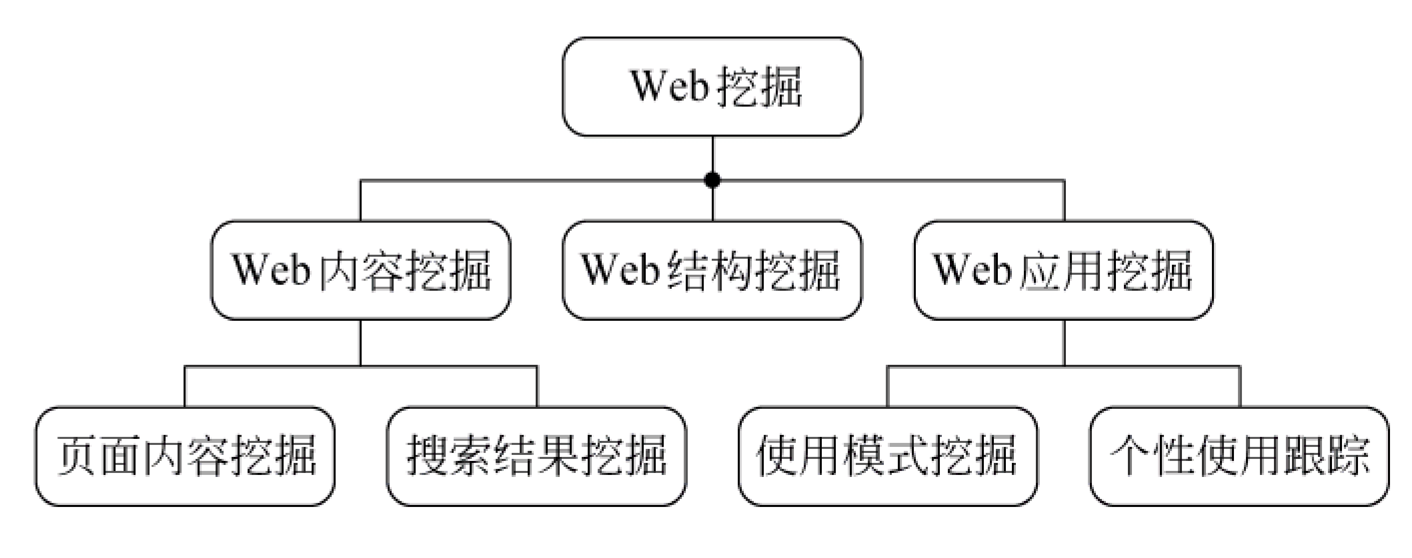


图 11.2　Web 挖掘分类

(1) Web 内容挖掘

内容挖掘是用来提取文字、图片或其他组成网页内容成分的信息和知识。哪个站点卖汽车？哪些页面是中文的？哪些页面是介绍音乐的，或是介绍新闻的？搜索引擎、智能代理和一些推荐引擎都使用内容挖掘来帮助客户在浩瀚的网络空间中寻找所需的内容。

Web 内容挖掘有两种策略：页面文本内容挖掘；对搜索引擎的查询结果进行进一步的处理，得到更为精确和有用的信息。

(2) Web 结构挖掘

结构挖掘是用来提取网络的拓扑信息，即网页之间的链接信息。从 WWW 的组织结构和链接关系中挖掘知识。哪些页面被其他页面所链接？哪些页面指向了其他页面？哪些页面的集合构成了一个独立的整体？可以对页面进行排序，发现重要的页面。

(3) Web 应用挖掘

应用挖掘是用来提取关于客户如何运用浏览器浏览和使用页面链接的信息。从 Web 的访问记录中抽取感兴趣的模式。例如，客户访问了哪些页面？在每一页上待了多长时间？下一步单击了什么？在站点中是按照怎样的访问路线进入和退出的？

WWW 中的每个服务器都保留了访问日志(Web access log)，记录了关于用户访问和交互的信息。分析这些数据可以帮助理解用户的行为，从而改进站点的结构，或为用户提供个性化的服务。

这方面的研究主要有两个方向：一般使用模式的挖掘和个性化使用记录的追踪。一般使用模式的挖掘，通过分析使用记录来了解用户的使用模式和倾向，以改进站点的组织结构；而个性化使用记录的追踪则倾向于分析单个用户的偏好，其目的是根据不同用户的访问模式，为每个用户提供定制的站点。

(4) 区别与联系

因特网是由许多用链接联系起来的网页组成。每个单独的页面都由多种成分组成，例如文本、图片及指向其他页面的链接等，网络服务器提供了对这些成分的访问权限。一个网页是由一些称为框架(frame)的结构组成的。

进行结构挖掘的原材料是一套将文档联系起来的超级链接。内容挖掘的原材料由那些存储于数以百万的文件中的文本组成，这些文件可以让任何客户通过内置 HTML 和 XML 标记的网络浏览器的一个按钮来访问。内容挖掘和结构挖掘都需要一种相对的静态的网络，也就是说，网页和链接要像静止在某个特定的时刻。对结构挖掘的理想的表达方式是用图形的方式(实际上是有向图，因为链接总是在一个方向上由一个网页指向另一个)。这种理想的图可以映射整个网络中链接所有文档的全部链接。内容挖掘的理想表达方式是一个索引。这个理想化的索引链接网络上每个网页中的每一个字符串、单词、短语、声音和图像。

结构挖掘和内容挖掘都不需要或提供有关客户行为的知识，结构挖掘提示了哪些页面通过当前页可以几步内到达，但并不关心多少人会实际用到这条通路。内容挖掘提示了网页的主题，但并不关心谁会真正阅读它。内容挖掘可以用于找出所有关于酒类的网页，而结构挖掘可以将这些网页组织成零售站点的聚类。对于葡萄酒的购买者和白酒的购买者的区别，就需要另一种类型的 Web 挖掘，即第三种称为应用挖掘的 Web 挖掘，它主要集中于挖掘客户的行为，特别是随着时间的变化。有时感兴趣的时间片很短，例如对于访问者在一次单独的会话中在一个站点中的访问路径的分析；在其他时候时间片又会比较长，例如对于在一个零售站点长期注册的购买者的购买行为的分析。

就像结构挖掘的理想图或内容挖掘的理想索引一样，可以想象一种应用挖掘的理想的数据表现形式。它可以是一个客户配置的知识库，并且可以不断地更新网络上每一个客户的配置。每个配置都会记录或描述某个单独的客户与网络的交互情况，包括所访问的站点、访问的路线、提出的问题、阅读的文档和购买的物品等。

比起内容挖掘的网络索引或结构挖掘的连接图，应用挖掘的理想表现形式实现起来要难得多。建立索引和图信息可以自由地从任何访问网络的客户那里自由获取，毕竟使文档和链接更加方便应用是网络技术的主要目的。比较而言，建立的客户信息是分散在各个网络日志、应用服务器日志、广告服务器日志、商业服务器日志、商品数据库和客户数据库中，它们分属于不同的组织，并且这些组织很多都不希望分享他们拥有的信息。因此，应用挖掘描述只能限定于描述访问者对于特定的站点的访问情况，或同一网络的站点。

结构挖掘、应用挖掘和内容挖掘都是 Web 挖掘的有价值的应用，它们完全可以被称为“对网络的挖掘”。

11.3.2 Web 内容挖掘

内容挖掘是从组成 WWW 的网页中提取信息的过程。内容挖掘最广为人知的一个应用是搜索引擎，没有它，网络将变得一无是处。Web 内容挖掘的基本技术是文本挖掘。

1. 信息检索

网络上有数不清的信息、留言，还有彻头彻尾的垃圾。找到需要的信息是一件不太容易的搜索工作，因为对于大多数的主题来说，网络只是一个“贫矿”。如果在网络上的所有文件

都被明确地标记了关键词或是可以清楚地描述文章内容的元数据，客户可以向图书馆管理员那样使用搜索引擎，搜索起来就不需要那么复杂的算法，只要简单的查询就行了。

网页正从 HTML 向 XML 转变，这样一来，可被利用的结构化元数据将大大增加（HTML 只是从文档显示的方式这一角度出发来描述文件的标准；XML 是一个扩展的标准，它可以让使用者通过约定的标记来表达语义上的信息）。现在大多数文件都没有元数据信息，必须要参照内容。数据挖掘在信息检索中的困难在于创造元数据去完成查询，比如“关于高血压的疗法”，要找到相关主题的网页（从这点来说，网络上很多的声音、图片资源都不能被查询到。因为现有的“内容挖掘”都是“文本挖掘”。虽然对于我们来说，区别不同的圆舞曲的调子、不同的图片是很容易的，但让计算机从中间提取出这些特点的算法，现在并没有被广泛地使用）。

信息检索的目标是找到想要找的，而不理会其他。这个想法可以由研究者从两个方面来判断该查询的有效性：“召回（recall）”和“精度（precision）”。“精度”回答了“在返回的网页中，正确的标题的比例是多少”的问题；“召回”则是回答“在所有正确网页中，返回了多少”的问题。这两个目标在某种程度上说是矛盾的。一个搜索引擎针对任何一个请求返回所有的网页，可以说有了很高的“召回”，但是只有很低的“精度”；反之，只返回一个正确主题网页的搜索引擎，可以说有很高的“精度”，但“召回”很低。

“召回”和“精度”哪个更重要？要看查询的性质。一些问题可以在查找到的一个网页里轻易地得到回答，有些则要参照很多网页。

搜索引擎努力地提高“精度”和“召回”数量，这两者都依靠于按主题分类的能力——这也是一个数据挖掘中十分吸引人的挑战。

2. 基于内容的分类

分类是数据挖掘中常见的一类事情。市场活动中期望针对目标将人群进行分类；信用卡交易被分类为风险种类；网页被标记可以区别它们书写语言的标签。在所有的例子中，都有一个主要的种类列表，每一个新的观察材料必须被分到一个合适的类别中。在内容挖掘中，分类的任务通常精简成为网页关键词。当然，一个页面可以被许多关键词所描述，这些关键词可以被分配不同的可信度。

一个有用的内容分类是决定文档以什么语言写成。语言信息可以被用来限制搜索结果，或是以客户可以读懂的语言返回结果。

许多数据挖掘技术可以用来分类。但是，其中大多数依赖于在数据库存放的记录的数据结构，在这样的记录中每个记录都有相同的字段。大多数文本型的数据都是非结构化的。在“填空”形式的表单中，虽然内容都是由客户填写的，但是每个填空中的字符都是存放到了数据库中特定的字段中，比如“姓名”或是“职业”。答案的内容经常是由一个下拉框给出。每个教师都知道多选题比简答题容易打分。但是，简答题更能考出学生的水平。在内容挖掘中面对的是同样的问题。结构化的内容利用标准的算法容易分类，但是客户真正感兴趣的材料却是非结构化的文本。

“k 最近邻（k nearest neighbo k-NN）”，这种方法很好地适用于在网页中利用关键词进行聚类。在 k-NN 方法中，每个新的网页与在数据库中预先聚类的例子进行对比，新网页将

出现和一些现有的网页非常类似，与另一些非常不同的情况。通过使用 k-NN 可以对相同的网页进行聚类。相似度越高，聚类的可信度也就越高。

两个网页类似具体是指什么？这需要一个函数，给出两个网页，返回一个用来描述它们有多少相似的数值。“类似”的概念可以类比“距离”这个概念。在物理空间中，物体的位置是由它在某一轴线上相对原点的距离所决定的。若知道两个点的 x、y、z 坐标（或经度、纬度、高度），就可以通过在轴线上的距离用简单的勾股定理来得到它们之间的距离。从概念上来说，两个类似的文本之间的距离是指，特定的一些维度，如客观物质、等级、幽默度等。依靠上下文，不同的维度可以有不同的权重。在有些目的下，两首打油诗虽然主题不同，创作语言不同，也可能是“类似”的。在另一些上下文中，一首古老的船歌、绿色和平组织的筹款信都会被认为类似的，因为它们都和捕鲸有关。

遗憾的是，这种确定文本差异的方法难于自动实现，因为它依赖于对内容的理解。当人们都被要求去比较两张网页的时候，总是从试着理解内容开始，计算机可不会这么干。

研究人员已经提出了用于测量两段文本之间“距离”的方法，即依照它们所含有的“字”本身而不需要理解这个字的含义，这种技术可以用数学函数表达。这个函数充分考虑了常用字的数量，还有一些冷僻的词汇。一旦有了可以应用于文本且返回结果的函数，一套新工具的出现就可以说有眉目了。

两个基于相同主题、用同一种语言（当然用的是一套词汇）的网页，可以说比较接近。一个作者的两部作品通常也是比较类似的。这样一来，可以从一个著名作家的作品中选取一段和一个有争议的作家的作品进行比较，判断出这段作品的原作到底是谁。

这种公开辩论的文本型挖掘已经被 Vassar 大学的 Don Foster 教授在确定有争议文件上所使用，并取得了引人注目的成功。

3. 从纯文本中提取信息

内容挖掘的一个目的就是从纯文本中得到有用的信息。要达到这样的程度，就必须真正地理解文本，而这还没有达到。但是在一个有限制的范围内，识别出一些特定的信息是可能的。那些追求信息提取的研究者的一个希望就是通过将纯文本转化为结构化的数据，能够直接应用数据挖掘技术，从而作出预测。这种从非结构化数据中创建结构化数据的过程叫做特征抽取。

特征抽取在网络上的应用是作为购物的一个辅助工具。有许多这样的服务，这些服务是寻找电子商务的站点并比较相同商品的价格。这要求它要有识别出两个站点正在销售同一商品（网站在卖着某些东西）的能力。

但是这种服务的工作效果并不是很好。我们认为对于这种比较购物的问题的解决，可能通过以 XML 标记的形式向网站中添加更结构化的内容比通过提高从非结构化的文本中提取信息的技术来得更加迅速一些。

11.3.3 Web 结构挖掘

结构挖掘可以告知一些站点的受欢迎程度和它与其他站点的距离（通过跳转次数来判定）。进一步，可以通过查看一个单独站点的网页的链接情况及相互链接的情况来学习其内

部结构。

网络的总体结构是十分迷人的。一个对于网络的分析将提示出人类分为数个不同的语言群落,并且任何以某种语言写成的页面总是链接与它相同语言的页面。

万维网(WWW)是一个有向图 $G=(V,E)$,V 是页面的集合,E 是页面之间的超链接集合。页面抽象为图中的顶点,而页面之间的超链接抽象为图中的有向边。顶点 v 的入边表示对 v 的引用,出边表示 v 引用了其他的页面。所以 Web 页面之间的超链接揭示了 Web 结构。

每个网页是这个图的一个结点,每个链接是一条边。之所以这个图是有向的,是因为存在由 A 指向 B 的链接并不等于也存在 B 指向 A 的对应链接。一个站点 A,它的每一个网页都包含了一个指向主页的链接。大部分的链接都是站内的,也可以指向站外的网页。

1. 网页的引用

在"不是出版,就是毁灭"的学术世界里,引用一直是保持成绩的一个方法。仅仅是出版过文章是不够的,重要的是其他人的确读过它们并且觉得它们有用。一篇文章的有用与否在于这篇文章出现在其他文章的参考书目中的次数。特别是作者,会因为他的作品的重复引用而在某个学科出名。

原则上讲,网络这种全球性结构也以同样的方式使网站保持成绩。通向这个站点链接越多,它就一定越重要。实际上对于站点管理者来讲,得到一个关于所有链接的准确视图是非常困难的,因为网络的结构绝不是静态的。被各大搜索站点用于建立索引的"网络爬行者"(Web crawler)是最易得到这种信息的来源。

如果没有人点击它们,静态链接就显得不是特别有用,就像一个科学家,如果没有人读他的著作,他就不是权威一样。从对 www. data-miners. com 网络日志的分析中,令人惊讶地得知,几乎没有人是从其他站点的静态链接到这里来的。有 47%的浏览者是直接在浏览器中键入的网址,或是收藏有该站点,或是将 www. data-miners. com 作为它们的默认开始页面。

当指引人们去浏览数据挖掘网站的时候,并不是所有的搜索引擎都是平等的。非常明显,超过 33%的通向该站点的网络搜索都是来自于 Google 的。相反地,仅仅有 3%的客户是通过 AltaVista 这个搜索引擎找到的。

为什么是 Google 能够更好地指引人们来到 Data-Miners 公司?

答案是,与不同的搜索引擎决定什么页面能够吸引读者的方式有关。AltaVista 是基于内容挖掘的,而 Google 同时还使用了结构挖掘。总之,当一个客户输入要搜索的字串"数据挖掘查询",AltaVista 将很高兴地返回任何谈到数据挖掘查询的页面,而 Google 则根据对于有关该题目链接的页面的结构分析来返回它认为是权威的页面。

网页引用的 Page-rank 方法是 Brin 和 Page 于 1998 年提出的一种方法。假设要搜索某一给定话题的 Web 页面,例如金融投资方面的页面。这时除了希望得到与之相关的 Web 页面外,还希望所检索到的页面具有较高质量和权威性。权威性(authority)可由 Web 页面链接来反映。Web 不仅由页面组成,而且还包含了从一个页面指向另一个页面的超链接。超链接包含了大量人类潜在的语义,它有助于自动分析出权威性语义。当一个 Web 页面的

作者建立指向另一个页面的指针时,可以看作是作者对另一页面的注解。把对一个页面的来自不同作者的注解收集起来,就可以用来反映该页面的重要性,并可以很自然地用于 Web 页面权威性的发现。可见,大量的 Web 链接信息提供了丰富的关于 Web 内容相关性、质量和结构方面的信息,这对 Web 挖掘是可以利用的一个重要资源。

Page-rank 的基本思想是:

(1) 一个页面被多次引用,则这个页面很可能是重要的;

(2) 一个页面尽管没有被多次引用,但被一个重要页面引用,则这个页面很可能是重要的;

(3) 一个页面的重要性被均分并被传递到它所引用的页面。

2. 中枢和权威

要在庞大的满足条件的文档中找到最有趣的或最权威的文档是非常困难的。

康奈尔大学的 Jon Kleinberg 提出了一种被广泛采用的技术来解决这个问题。他的想法是利用这样的事实,在建立从一个站点到另一个站点的链接时,网站的管理者将会对将要建立链接的网站的价值作一个判断。每个到站点的链接对这个站点都是有意义的。久而久之,那些决定给同一目标站点提供链接的站点能够证实目标的权威性。进一步,所要链接的站点的可靠性也可以通过它们链接到的站点的权威性来判断。一个拥有许多其他好站点推荐的站点可以用来决定另一个站点的权威性。

Kleinberg 提出一个链接到许多权威站点的站点叫做中枢(hub);被许多中枢链接的站点叫做权威(authority)。这两个概念放在一起可以辨别出权威和大众化站点(如 Yahoo)之间的区别。一种寻找权威的结构化的方法就是,用其他的站点到该站点的链接数来将它们分级。要给站点分级,不要用指向它们的链接的总数,而是用指向它们的标题相关的中枢的数量来分级。

结构挖掘是为提取信息而对网站的链接进行分析的过程,对单一网站局部结构的分析,对于理解此网站的创办的目的和设计很有帮助。对全局结构的分析是一种将一个网站分解成多个紧密联系的子网站的途径。运用全局结构挖掘,有可能把网页归类为中枢(到许多其他网页的很好的跳板网页)和权威(许多网页设计师都觉得值得链接到的网页)。

hub/authority 方法是 Kleinberg 于 1998 年提出的。基于商业或竞争的考虑,很少有 Web 页面会指向其竞争对手页面。例如,可口可乐不会链接到其竞争对手百事可乐的 Web 页面。这些现象使 Web 链接结构存在一些局限性。

为此人们提出了另外一种重要的 Web 页面,即中枢页面。一个 hub 是指一个或多个 Web 页面,它提供了指向权威页面的链接集合。hub 页面本身可能并不突出,或者说可能没有几个链接指向它们。但是,hub 页面却提供了指向某个公共话题最为突出的站点链接。此类页面可以是主页上的推荐链接列表,例如一门课程主页上的推荐参考文献站点,或商业站点上的相关信息站点。hub 页面起到了隐含说明某权威页面的作用。通常,好的 hub 是指向许多好的权威的页面;好的权威是指由许多好的 hub 所指向的页面。这种 hub 与 authority 之间的相互作用,可用于权威页面的挖掘和高质量 Web 结构及资源的自动发现。这就是 hub/authority 方法的基本思想。

3. 导航页

导航页的存在主要为了链接其他页面。客户不必在导航页上花费太多的时间，却会频繁地到这个导航页面上。对客户来说，导航页能够很容易地找到客户想要找的网页。通过比较从入口到目标网页所要求的点击数和浏览者平均的点击数，会得到一些关于怎样设计好的网络站点和怎样链接网页的建议。

4. 目标页

浏览者通常花费大量的时间在目标页上。这一网页实际上给浏览者提供所要查找的信息、娱乐和商品。总之，目标页给浏览者提供所有的内容。

目标页一般是固定的。当浏览者在一个目标页上花费了大量的时间时，希望这是因为找到了他们所需要的东西。当然，并不是所有的浏览者都是这样的。或许他们有许多的疑惑，或者要求查到更多的东西，要么由于其他的原因使他们的输入速度非常慢，从而导致了他们在此网页花费了大量的时间。通常仔细分析登录数据，可以得出他们的不同之处。要指出的重要的一点是：如果没有应用数据的配合，一个网站的静态结构是没有很大用处的。应用数据允许比较这个网站的结构，因为它反映了设计者的思想，也就是说反映的是这个网站及其实际的行为数据该如何使用。

5. 形成功能

某个网站的局部结构很大程度上依赖于它的用途。网站有许多不同的模型。一个零售站点可能都是以同样方式列出商品页面，并且建立了一个存储在关系数据库中的商品和价格的桥梁。有一些会模拟离线资源，如报纸或是杂志。其他的则包含可构建的会话，这些会话能够定制并能以多种方式排列，满足特殊客户的需求，这种站点诸如 yahoo. com，甚至一个非常简单的网站，如 www. data-miners. com，都是由它的目标来决定结构。该站点的基本目标是：

（1）让寻找数据挖掘顾问的人们找到站点并联系站点。

（2）允许搜索数据挖掘教程和研讨会的人注册成为站点的会员。

（3）让搜索有关数据挖掘的书籍的人们能够购买站点的书。

这意味着可通过一个单击从首页到达联系信息、当前课程表、课程注册链接和售书链接。想要实现更多目标的站点的结构就会更为复杂，但是基本的原则仍然是使得浏览者容易做想让他们做的事情。

11.3.4 Web 应用挖掘

对于一个链接或网页来说，一个有用的属性是它的大众化程度，这是用在给定的时间内访问它的客户的数量来衡量的。在大多数情况下，当考虑的是对客户的理解时，应用模式就是非常关键的。应用模式可以从多个层次检测和挖掘到，从单个客户在一次对话中的一系列的单击到跨越了几个月或数年的客户群的购买模式中。通常，长期以来收集的信息可以组成一个特性文件，依次提供当前客户的快照。这些特性文件可以被用来产生建议和个性

化服务。

应用挖掘有很多应用,从提高网站的设计到改善客户关系的管理。随着人们需求的不断增长,所要求的数据资源也更加丰富多变。

1. 点击流分析

用于 Web 挖掘有效的最简单的数据就是点击流——由一个站点的网络服务器来接受的网页请求。点击流的定义是一个网站浏览者通过点击链接所明确要求的一系列文件。

点击流数据类似于超市的扫描数据。在超市,每个售货系统为每个扫描过的商品建立一个记录。这些记录在进行数据挖掘之前都是列入市场购物车的。一旦市场购物车经过鉴定,就能够提出商品分级的问题,如哪些商品放在一起卖比较好。要想从商品级转到客户级问题,市场购物车必须在一定程度上与购物者联系起来,这样才能使模式和行为被检测出来。

在网络世界里存在着类似的情况。在最低层次中,记录了所有客户的浏览器所请求的 JPEG 和 HTML 文件。这些点(hits)必须归入页面视图(page views)中。在找出一个链接客户和会话的方法之前是没办法得知各个客户的行为的。

从页面视图层移向单一的客户层,尽管得到很少的数据,但这却更为有趣。最有趣的分析是在单个客户层追寻的行为达数个星期甚至数月,或是在会话层的一次浏览的过程中追寻客户的行为。每个单一的客户可能会有很多对话。每个对话可能会有许多的页面视图,并且每个页面视图都可能被网络日志记录为许多的点。

2. 网络日志

点击流的分析始于网络日志。虽然一个网络站点看起来似乎是由许多网页组成的,但实际上网站注册所遵循的还有点不同。也就是说,个别细节需受网络浏览器控制,有所不同的理由是:作为一个独立的网页,浏览窗口是一种既普通又复杂的事物,包括各种会话框,每一个会话框都会显示内容不同的超文本标记语言标准文档,每一个这种文档又依次含有代表图片文档的附注内容。为了便于在浏览窗口中制作网页,每一个这样的文档都受服务器控制,以便及时把请求有效地传送给服务器。

当客户单击单独的网页时,为了一个组成对象,单个的请求会转为多个单击,因为它是由不同的服务器来操作的,这些单击也将通过各种指标予以记录,因为服务器已获得成千上万个与某个特殊的网页视图有关的其他浏览器的单击。

3. 应用日志

要使得一个网站浏览者全面地了解,必须要求从应用服务器上得到的数据是完整的点击流数据。

在一个现代电子商务的体系结构中,知道什么网址被请求并不能说明什么问题,因为网页的内容都是由一个应用服务器通过空白页建立的。总而言之,同样的 URL 地址在不同的时刻可能就代表了不同的含义。应用服务器有详细的关于什么被请求和它代表了什么的信息。

只有应用日志知道什么时候一些商品放在商店里，什么时候该拿走，什么时候客户进来，什么时候客户出去。这是商品制造商和销售商感兴趣，而不是网站设计者感兴趣的客户行为。这也是来自网络的数据和来自其他渠道的数据，如商店、电话销售中心等。

4. 日志挖掘的基本流程

对 Web 访问日志（Web log）进行分析和挖掘要经过一系列的工作。

基本的流程包括如下步骤：

（1）首先要对 Web log 进行清洗、过滤和转换，从中抽取感兴趣的数据。

（2）将资源的类型、资源的大小、请求的时间、在资源上停留的时间、请求者的 Internet 域名、用户、服务器状态作为数据立方体（data cube）的维变量，将对页面和文件请求次数、来自不同 Internet 域请求次数、事件、会话、错误次数分别作为在这些维变量下的度量变量建立数据立方体（data cube）。通过对 data cube 的切块、切片分析可以回答：哪些成分或特色被经常或偶尔使用，网络流量随时间的变化规律（按时、日、月等），用户在不同 Internet 域的分布情况，来自不同地区的用户在存取方式上是否有差异。

（3）利用成熟的数据挖掘技术（如特征提取、分类、关联、预测、时间序列分析、趋势分析）进行 Web 流量分析、典型的事件序列和用户行为模式分析、事务分析，可以回答成分和特色在哪些上下文中被使用；什么是典型的事件序列；在用户中有共同的行为模式是什么；不同用户群在使用和行为上有什么差异；用户的行为是否随时间变化，以及怎么变化等问题。

通过分析 Web 访问日志能帮助理解用户的行为和 Web 结构，因此，可以改进 Web 页面的设计和 Web 应用程序，发现潜在的电子商务客户。

OLAP 从不同的视角、不同的概念层次提供了数据视图，而 Web log 数据挖掘提供了深层次的报告，像时间序列分析、相关、分类等。通过使用这类 Web log 文件，可以进行一些研究工作，如系统性能分析，通过 Web 缓存改进系统设计、Web 页面预取、Web 页面交换（swapping）；认识 Web 信息访问的本质；理解用户的反映和动机。例如，有些研究提出了可适应站点（adaptive site）的概念，即可以通过用户访问模式的学习，改进其自身的 Web 站点。

Web log 分析还有助于建立针对个体的个性化 Web 服务。由于 Web log 数据提供了用户访问 Web 页面的信息，因此 Web log 信息可以与 Web 内容挖掘和 Web 结构挖掘集成起来，用于 Web 页面的等级划分、Web 文本的分类和多层次 Web 信息库的构造。

5. 应用挖掘提高网站可用性

通常在一个网站挖掘使用数据的原因在于要提高这个网站的可用性。分析的第一步是收集客户的使用途径。每一个客户的会话都是一系列的网页要求。一般来说，每个商店每次的订货之间都有着特定的目的性。这些网页请求的联系可以由分析商店业务之间的联系而得到。这些联系暗示这些网页之间要添加一些附加链接。如果在一个会话中经常被访问的两个页面彼此之间没有链接，如果加上链接，客户会感到更加方便。在一次网络购物中，站点并不关心客户以怎么样的顺序买东西。在网站访问时，页面访问的顺序非常重要，以至

于站点要把这个顺序作为一个整体来研究。当关联规则应用到这个序列时,就可以得到这次业务的顺序规则,比如先到主页,再找工作列表,然后到联系方式。

这样的会话可以依照不同情况分为不同的类。这些类代表了不同的客户,比如老客户和新客户;浏览的客户和想购物的客户。不同的访问者访问同一个网站的目的是不一样的,比如,访问一个零售商网站的人的目的可能是购物或寻找就业机会。

所有这些都应该轻松完成,但是任何尝试过的人都知道情况并不是如此。应用挖掘可以帮助确定使用模式并且对使用提出改进的建议。比如,一个报社发现那些关心报纸递送业务的客户要浏览 7~8 个网页才能完成订阅递送,通过减少页面,减至 3 个,其结果是成倍地增加了那些从网络订阅报纸递送的客户的数量。

习　题

1. 文本挖掘的概念是什么?
2. 文本挖掘与数据挖掘有什么不同?
3. 文本挖掘的主要任务是什么?
4. 文本特征包含什么内容?
5. 如何形式化表示文本特征?
6. 文本特征提取的基本算法过程是什么?
7. 说明文本挖掘的功能层次内容。
8. 说明文本关联分析的基本思想。
9. 文本的层次聚类法的基本过程是什么?
10. 说明平面划分法的基本思想,它与层次聚类法的区别在什么地方?
11. 说明文本关联分析方法的处理过程。
12. Web 信息有什么特点?
13. Web 挖掘与文本挖掘有什么区别和联系?
14. 说明 Web 挖掘的分类。
15. Web 内容挖掘的含义是什么?
16. Web 结构挖掘的含义是什么?
17. Web 应用挖掘的含义是什么?
18. “召回”与“精度”的含义是什么?它们之间的关系是什么?
19. 什么是基于内容的分类?
20. 如何从纯文本中提取信息?
21. 页面引用的 Page-rank 方法的基本思想是什么?
22. 什么是中枢站点和权威站点?
23. hub/authority 方法的基本思想是什么?
24. 点击流的定义是什么?点击流数据中包含哪些内容?
25. 网络日志挖掘的基本流程是什么?

第 12 章　数据仓库与数据挖掘的发展

12.1　综合决策支持系统

12.1.1　从管理科学到决策支持系统

1. 管理科学

管理科学(management science, MS)的传统名字叫运筹学(operations research,OR)。

1940 年 9 月英国成立了由物理学家 P. M. S. 布莱克特领导的第一个运筹学小组。在第二次世界大战中,为开展反潜艇的侦察,以及组织有效的对敌轰炸等方面做了大量的研究。1947 年 G. B. 丹齐克提出线性规划及其通用解法——单纯形法。20 世纪 50 年代末,美国大企业在经营管理中大量应用运筹学,开始时主要用于制定生产计划,后来在物资储备、资源分配、设备更新、任务分派等方面应用和发展了许多新的方法和模型。这些研究推动了管理科学的发展,为决策提供科学的依据。

管理科学是对管理问题用定量分析方法,建立数学模型,通过求解计算,达到辅助管理决策的一门学科。管理科学是用数学模型方法研究经济、国防等部门在环境的约束条件下,合理调配人力、物力、财力等资源,通过模型的有效运算,来预测发展趋势,制定行动规划或优选可行方案。

模型是对客观规律的一般描述,人们通过对模型的认识来增强对付复杂的大规模问题的处理能力,使人们尽可能地按客观规律办事,不犯错误,取得预期的效果。例如,人口模型反映了人口发展的规律以及主要影响因素。通过人口模型的计算,为国家制定政策、控制人口的出生率提供辅助决策建议。

2. 决策支持系统

管理科学与运筹学是运用模型辅助决策,体现在单模型辅助决策上,模型所需要的数据在计算机中以文件形式存储。随着新技术的发展,所需要解决的问题会愈来愈复杂,所涉及的模型愈来愈多,不仅是几个而是十多个,几十个,以至上百个模型来解决一个大问题。这样,对多模型辅助决策问题,在决策支持系统出现之前是靠人来实现模型间的联合和协调。决策支持系统的出现是要解决由计算机自动组织和协调多模型的运行及数据库中大量数据的存取和处理,达到更高层次的辅助决策能力。决策支持系统的特点就是增加模型库和模型库管理系统,把众多的模型有效地组织和存储起来,并且建立了模型库和数据库的有机结合。这种有机结合适应人机交互功能,自然促使新型系统的出现,即 DSS 的出现。决策支持系统不同于 MIS 数据处理,也不同于单模型的数值计算,而是它们的有机集成。它既具

有数据处理功能又具有模型的数值计算功能。

决策支持系统的结构形式，如图 12.1 所示。

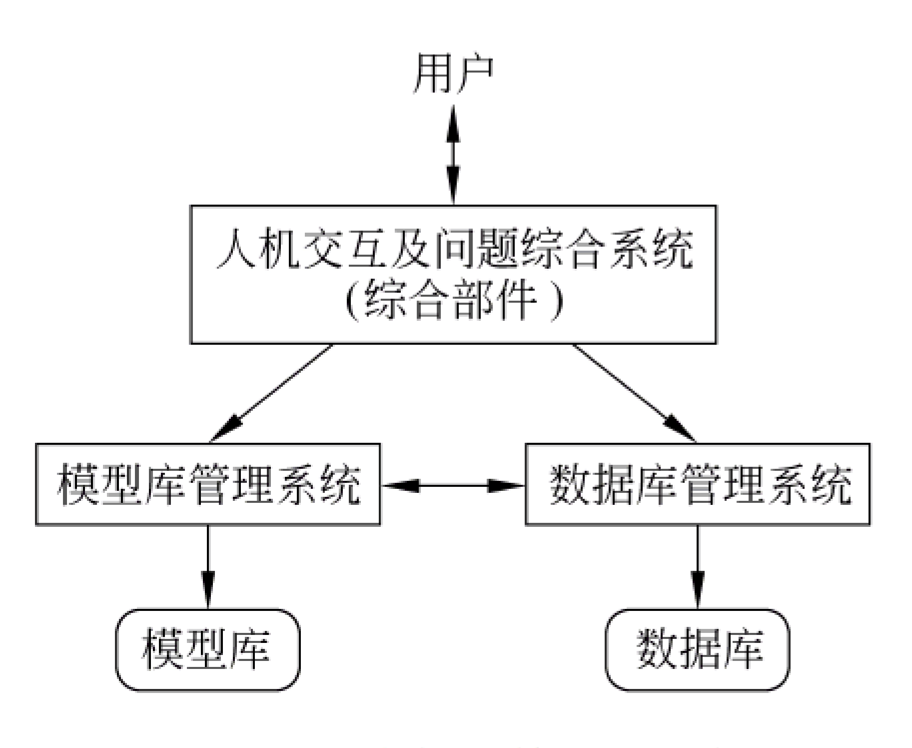


图 12.1 决策支持系统结构

“人机交互及问题综合系统（综合部件）”可理解为对实际决策问题的处理与人机交互的综合作用。

在决策支持系统出现之前，组合多模型辅助决策早已出现，具体做法是对各模型编制程序并在计算机中运行，模型之间的关联由人来完成，即由人来完成模型的组合。对模型间的数值计算和数据处理，只能由人在计算机外进行。因为每个模型本身是不考虑与其他模型之间的联结问题，这项工作只能由人来完成。在出现决策支持系统之后，这种模型间的处理，应由“人机交互及问题综合系统”部件来完成。解决了这个问题才能使多模型的组合运行能在计算机中自动进行。多模型的组合形成了系统的方案，能解决更复杂的问题，多模型组合的自动运行为改变方案中的模型和数据带来了方便。在系统方案中采用不同的模型或数据的组合形成不同的方案，故决策支持系统为解决半结构化问题（部分由计算机完成，部分由人来完成的问题）成为可能。

为达到决策支持系统有效自动地运行，它对语言系统的功能要求比较高，即它应具有调用模型运行能力、数据库存取能力、数值运算能力、数据处理能力、人机交互能力 5 种综合能力，称为决策支持系统语言，它不同于数值计算语言（如 FORTRAN、C 等），还要有很强的数据处理（数据库处理）能力。DSS 语言应是两类语言（数值计算语言和数据库语言）的综合。

决策支持系统语言是使原来不能在计算机上实现的问题，即多模型组合辅助决策问题（即半结构化问题）能在计算机帮助下完成。

可见，决策支持系统是技术进步的产物。

3. 智能决策支持系统

智能决策支持系统（intelligent decision support systems，IDSS）是决策支持系统与人工智能技术相结合的系统。

人工智能技术主要是以知识处理为主体，利用知识进行推理，完成人类定性分析的部分智能行为。人工智能技术融入决策支持系统后，使 DSS 在模型技术与数据处理技术的基础上，增加了知识推理技术，使 DSS 的定量分析和 AI 的定性分析结合起来，提高辅助决策和支持决策的能力。

智能决策支持系统是 DSS 的重要发展方向，中国在 20 世纪 90 年代初期形成了高潮，建立了不少智能决策支持系统，研究文献也大量涌现。

传统的决策支持系统是以模型技术和数据处理技术为基础发展起来的，以 1980 年 R. H. Sprague提出的三部件结构为典型代表。在该系统中模型部件（模型库与模型库管理系统）是主体。在该决策支持系统中加入知识部件（知识库、知识库管理系统与推理机）后，

形成了智能决策支持系统。

在这里要说明的是，知识部件中知识库管理系统完成的是对知识的查询、浏览、增加、删除、修改、维护等管理工作，而推理机完成对知识的推理。知识一般需要经过推理才能用于解决实际问题。实际上，知识推理是建立从初始概念到中间概念，最后到目标概念的推理链。例如，“咳嗽”、“发烧”是人的症状，初始概念经过推理得出该人是“肺炎”或“肺结核”的目标概念。得出目标概念以后，才能对“病”进行“治疗”。医疗知识是通用的，但对不同人的病症，经过推理之后，得出的“病名”是不同的。不同的“病名”，“治疗”的方法将不同，“肺炎”和“肺结核”的治疗是完全不同的。可以说，推理机在知识部件中是重要的组成部分，是使用知识的重要手段。可见，知识部件不同于模型部件和数据部件，由知识库、知识库管理系统和推理机三者组成。

智能决策支持系统是决策支持系统与人工智能技术结合的系统，原决策支持系统的“人机交互与问题综合系统”在此变为“问题综合与交互系统”更合适。

在 IDSS 结构中，模型库系统（模型库与模型库管理系统）和数据库系统（数据库与数据库管理系统）是 DSS 的基础。人工智能技术包括专家系统、神经网络、遗传算法、机器学习和自然语言理解等。其中：专家系统的核心是知识库和推理机；神经网络涉及样本库和网络权值库（知识库），神经网络的推理机是 MP 模型；遗传算法的核心是“选择、交叉、突变”3 个算子，可以看成是遗传算法的推理机，它处理的对象是群体，这是一个动态库；机器学习包括各种算法库，算法可以看成是一种推理，它对实例库进行算法操作获取知识；自然语言理解需要语言文法库（知识库），处理对象是语言文本，对语言文本的推理采用推导和归约两种方式。可见，这些人工智能技术可以概括为

$$\text{推理机} + \text{知识库}$$

智能决策支持系统的结构表示如图 12.2 所示。

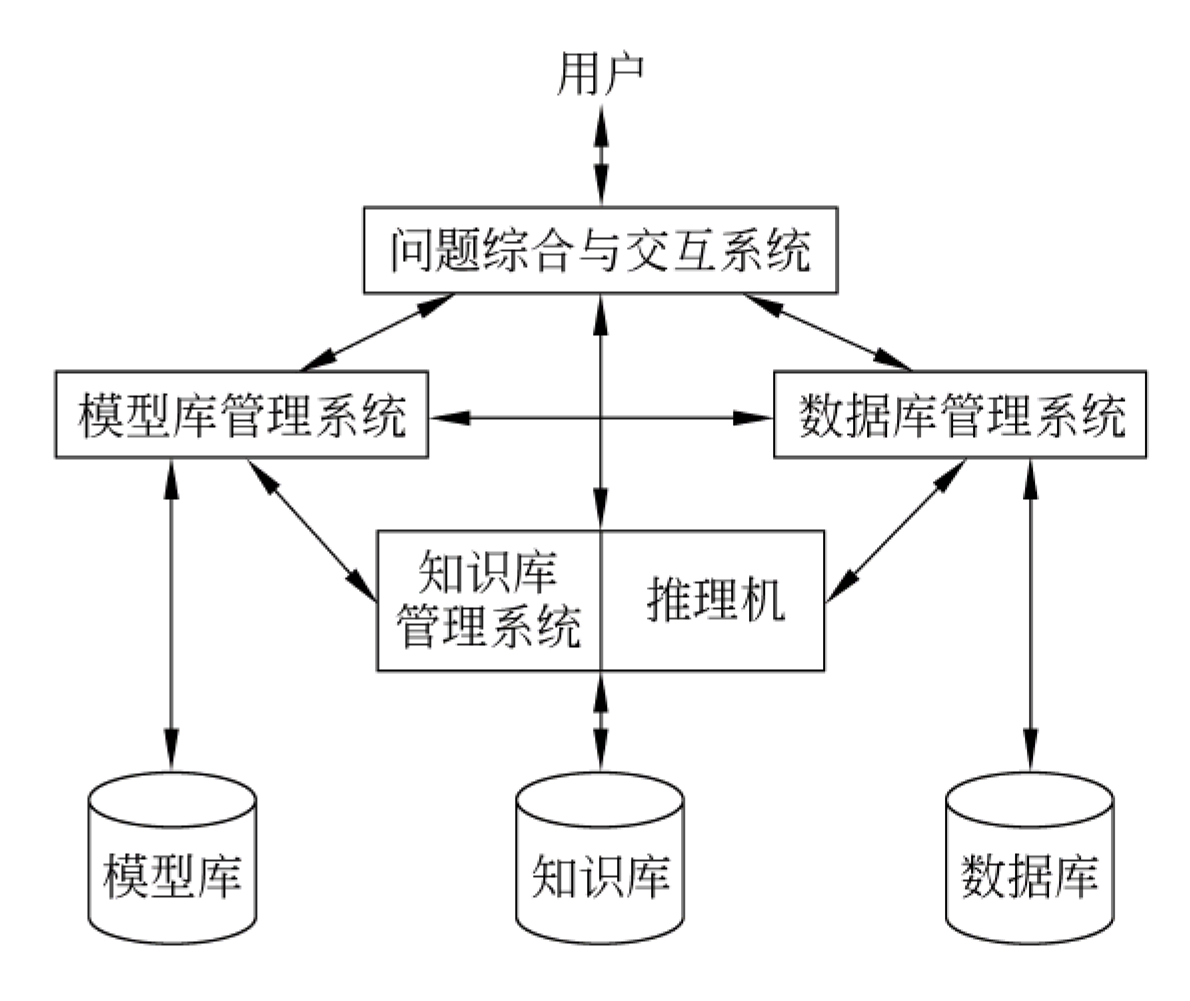


图 12.2　智能决策支持系统结构

智能决策支持系统中的人工智能技术种类较多，这些智能技术都是决策支持技术，它们可以独立开发出各自的智能系统，发挥各自的辅助决策作用。智能技术和决策支持系统结

合起来形成了智能决策支持系统。各种智能技术在智能决策支持系统中发挥的作用是不同的。一般智能决策支持系统中的智能技术只有一种或两种。

智能决策支持系统的特点是以模型计算和知识推理的方式辅助决策,称为传统决策支持系统。

12.1.2 基于数据仓库的决策支持系统与传统决策支持系统的结合

数据仓库是为辅助决策而建立的,单依靠数据仓库达到辅助决策的能力是有限的。数据仓库中有大量的综合数据,这些数据为决策者提供了综合信息,即反映企业或部门的宏观状况。数据仓库保存有大量历史数据,这些数据通过预测模型计算可以得到预测信息。

综合信息与预测信息是数据仓库所获得的辅助决策信息。

数据仓库中增加联机分析处理和数据挖掘等分析工具,能较大地提高辅助决策能力。联机分析处理对数据仓库中的数据进行多维数据分析,即多维数据的切片、切块、旋转、钻取等,只有通过分析更详细的数据,才能得到更深层中的信息和知识。数据挖掘技术能获取关联知识、时序知识、聚类知识、分类知识等。使用数据挖掘技术对数据仓库中的数据进行挖掘,才能获取更多的辅助决策信息和知识。

数据仓库和联机分析处理及数据挖掘结合的决策支持系统,是以数据仓库为基础的,称为基于数据仓库的决策支持系统,其结构如图 12.3 所示。

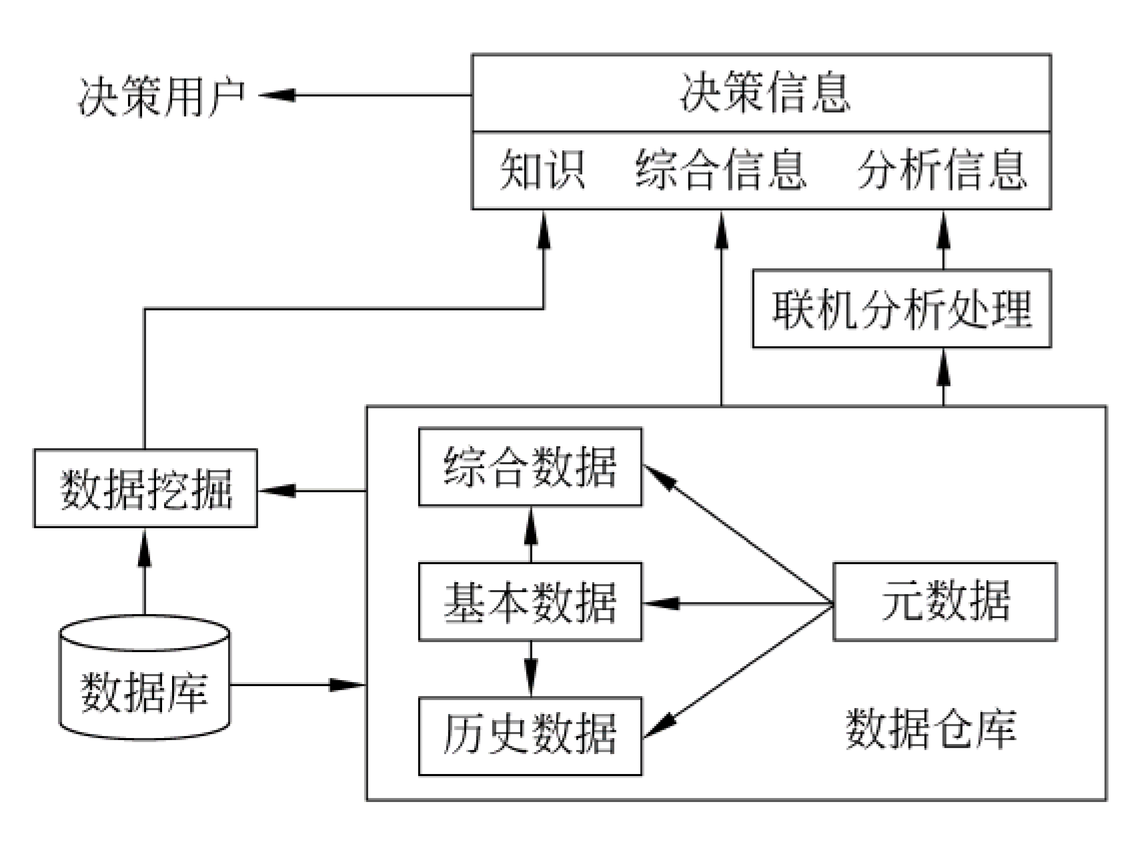


图 12.3 基于数据仓库的决策支持系统结构

概括地说,基于数据仓库的决策支持系统是从数据仓库的数据中获取辅助决策的信息和知识,为决策提供支持。

基于数据仓库的决策支持系统区别于 20 世纪 80 年代出现的基于模型的决策支持系统和 20 世纪 90 年代兴起的智能决策支持系统。基于模型和知识的智能决策支持系统是传统的决策支持系统,把基于数据仓库的决策支持系统称为新决策支持系统。

1. 新决策支持系统与传统决策支持系统的比较

新决策支持系统和传统决策支持系统几乎没有什么共同之处,它们是从不同的角度发展起来的,辅助决策的方式也不相同。由于两者不是覆盖关系,也就不存在相互代替的问

题，而是相互补充和相互结合的问题。

（1）新决策支持系统中数据挖掘获取的知识与传统决策支持系统的知识推理中的知识是不相同的。传统决策支持系统的知识来源于专家的领域知识和经验知识，而新决策支持系统的知识来源于数据仓库中的数据，它们的结合将扩大知识面。数据挖掘获取的知识也可用推理机来进行定性分析，也就是说，数据挖掘可以和专家系统结合起来。

（2）新决策支持系统中没有充分利用模型和模型组合来辅助决策。模型中的数学模型是管理科学/运筹学几十年来研究的成果，它们为各企事业单位的决策问题提供了广泛的辅助决策信息，取得了显著的决策效果。

新决策支持系统中联机分析处理主要是进行多维数据分析，通过切片、切块和钻取操作，可以找出问题出现的原因。

（3）决策支持系统的技术还没有完全成熟。传统决策支持系统虽然发展了二十多年，有很多研究成果，但没有完全成熟的产品，如模型库系统就是一个典型的例子。新决策支持系统刚发展起来，需要在实践中逐步完善。

传统决策支持系统和新决策支持系统结合起来，一方面可以相互促进、互相结合，对已成熟的技术可以先结合起来，逐步扩展到后成熟的技术。另一方面，这种结合为决策支持系统的发展前景指明了方向。

2. 新决策支持系统与传统决策支持系统的结合

将传统决策支持系统和新决策支持系统结合起来的决策支持系统称为综合决策支持系统（synthetic decision support system，SDSS）。

20 世纪 90 年代中期兴起的数据仓库是支持决策的新技术，数据仓库是从大量的数据中提取综合信息和预测信息进行辅助决策。它和传统决策支持系统有明显的区别。

紧跟数据仓库一起兴起的联机分析处理的数据组织是多维数据结构形式，它与数据仓库的数据组织是一致的。联机分析处理和多维数据分析的主要手段是对多维数据的切片、切块、旋转、钻取等操作。联机分析处理和数据仓库的结合提高了数据仓库的辅助决策能力。

数据挖掘技术也是 20 世纪 90 年代中期兴起的，它虽然是对数据库中数据的挖掘，但它应用于数据仓库后，在数据仓库中获取知识，也提高了数据仓库的辅助决策能力。

数据仓库与联机分析处理和数据挖掘三者结合起来，使辅助决策能力极大地提高，它们应用于实际决策问题而形成的决策支持系统是一种新型决策支持系统。这种新决策支持系统的典型特点是从数据中获取辅助决策信息和知识。它们以数据仓库中的大量数据为对象，数据仓库本身能提供综合信息和预测信息；联机分析处理提供多维数据分析信息；数据挖掘提供所获取的信息和知识，共同为实际决策问题辅助决策。

新决策支持系统不同于传统决策支持系统。传统决策支持系统是以模型和知识为决策资源，通过模型的计算和知识推理为实际决策问题辅助决策。传统决策支持系统是组合模型辅助决策的。大量的模型存放在模型库中，模型与模型间的连接是通过数据完成的，模型之间的连接数据一定是共享数据，它必须存放在数据库中。早期的决策支持系统中包含模

型库系统和数据库系统,是为实现多模型组合需要。模型的计算属于数值计算,组合模型的辅助决策完成了定量分析辅助决策效果。

知识推理是人工智能技术,以专家系统为代表的知识推理完成了定性分析辅助决策。早期的决策支持系统和专家系统的结合形成了智能决策支持系统,实现了定量分析辅助决策与定性分析辅助决策的结合,即达到更高的辅助决策效果。

新决策支持系统与传统决策支持系统在本质上是不一样的,也就是说,不能用新决策支持系统来代替传统决策支持系统。为了更有效地辅助决策,应该将新决策支持系统和传统决策支持系统结合起来。

3. 综合决策支持系统结构

把数据仓库、联机分析处理、数据挖掘、模型库(MB)、数据库、知识库(KB)结合起来形成的综合决策支持系统是更高级形式的决策支持系统。其中数据仓库能够实现对决策主题数据的存储和综合以及时间趋势分析。联机分析处理实现多维数据分析,数据挖掘从数据库和数据仓库中获取信息和知识,模型库实现多个模型的组合辅助决策,数据库为辅助决策提供数据,知识库中知识通过推理进行定性分析。它们集成的综合决策支持系统(SDSS)将相互补充和依赖,发挥各自的辅助决策优势,实现更有效的辅助决策。

综合的决策支持系统(SDSS)结构如图12.4所示。

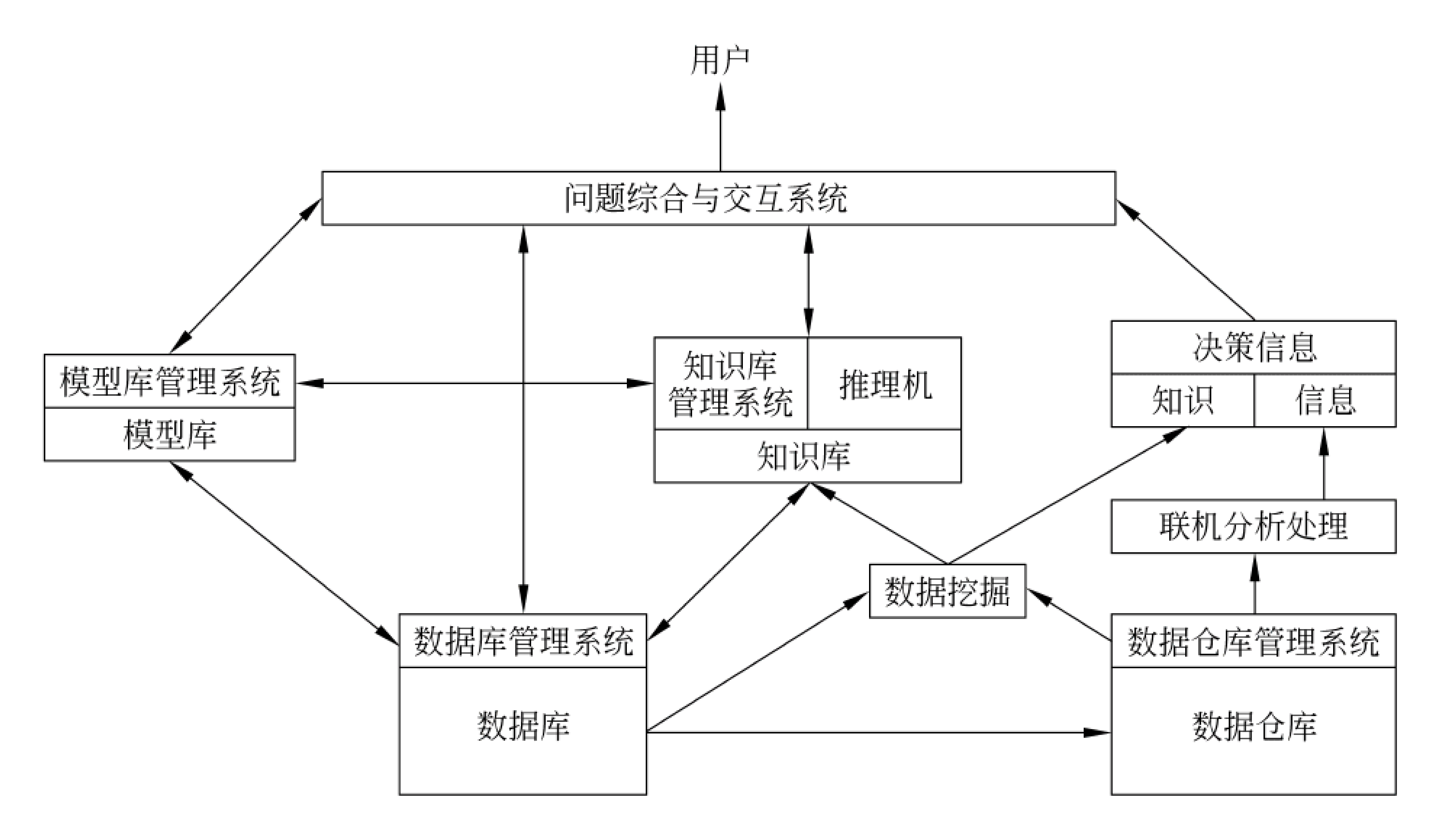


图12.4　综合决策支持系统(SDSS)结构

综合决策支持系统由3个主体组成:

(1) 模型库系统和数据库系统结合的主体。该主体完成多模型的组合与大量共享数据的处理,是利用模型资源辅助决策的。

(2) 数据仓库系统与联机分析处理结合的主体。该主体完成对数据仓库中数据的综合、预测和多维数据分析,是利用数据资源辅助决策的。

(3) 知识库系统(知识库、推理机和知识库管理系统)与数据挖掘结合的主体。该主体

完成知识推理,是利用知识资源辅助决策的。

12.1.3 综合决策支持系统发展趋势

1. 综合决策支持系统的兴起

数据仓库强调数据驱动,即以数据为基础,将传统的数据库系统的数据进行从面向应用的需求转变到数据仓库的面向分析的需求,向用户提供更准确和更有用的决策信息。但是数据仓库未明确提出利用模型的问题。

实际上,从数据仓库的结构图中可以看出,要完成从当前基本数据层中的数据汇总到轻度综合数据,再从轻度综合数据汇总到高度综合数据,是需要通过汇总模型来完成的。另外,数据仓库从历史数据中得到预测信息,是需要通过预测模型来完成的。可见,数据仓库达到辅助决策的目的仍需要模型。不过,数据仓库中使用的模型是固定和单一的,相对于数据仓库中的数据来说是次要的。

随着数据仓库的广泛应用和发展,数据仓库在逐步增加各种模型,来提高辅助决策效果。

以客户为中心的银行数据仓库使用模型情况如下:

(1) 分销渠道的分析模型

通过客户、渠道、产品或服务三者之间的关系,了解客户的购买行为、客户和渠道对业务收入的贡献、哪些客户比较喜好由什么渠道在何时和银行打交道、目前的分销渠道的服务能力如何、需要增加哪些分销渠道才能达到预期的服务水平。

为此,银行需要建立客户购买倾向模型和渠道喜好模型等。

(2) 客户利润贡献度模型

通过该模型能了解每一位客户对银行的总利润贡献度,银行可以依客户的利润贡献度安排合适的分销渠道提供服务和销售;知道哪些有利润的客户需要留住,采用什么方法留住客户;交叉销售改善客户的利润贡献度;哪些客户应该争取,完成个性化服务。另外,银行可以模拟和预测新产品对银行的利润贡献度,或者新政策对银行将产生什么样的财务影响,或者客户流失或留住对银行的整体利润的影响。

(3) 客户关系(信用)优化模型

银行对客户的每一笔交易中,知道客户需要什么产品或服务,例如,定期存款是希望退休养老使用;申请信用卡需要现金消费;询问放贷利息需要住房贷款等,这些都是银行提供产品或服务最好的时机。银行需要将账号每天发生的交易明细,以实时或定时方式加载到数据仓库中,校对客户行为的变化。当有上述变化时,通过模型计算,主动地与客户沟通并进行交叉销售,达到留住客户和增加利润的目标。

(4) 风险评估模型

模拟风险和利润间的关系,建立风险评估的数学模型,在满足高利润、低风险客户需求的前提下,达到银行收益的极大化。

银行通过以上模型实现以客户为中心的数据仓库决策支持系统,才能真正实现个性化

服务，提高银行竞争优势。

传统决策支持系统的优化模型为企业或者部门的发展提出了有效的途径。若数据仓库中能加入类似于优化模型的数学模型，将会提高数据仓库辅助决策能力。

在数据仓库系统中增加模型库来提高辅助决策效果，这就形成了综合决策支持系统的初型。

2. 网络环境的综合决策支持系统

Internet 技术推动了决策支持系统的发展。网络上的数据库服务器使数据库系统从单一的本地服务上升为网络上的远程服务，而且能对远地多个用户的不同客户机，同时并发地提供服务。新发展起来的数据仓库也是以服务器形式在网络上提供共享和并发服务。数据库和数据仓库都是数据资源。同样，将模型资源和知识资源也以服务器的形式在网络上为远地的客户机提供并发和共享的模型服务和知识服务。

模型服务器中可以集成大量的数学模型、数据处理模型、人机交互的多媒体模型等，为用户提供不同类型的模型服务，也可以为用户提供组合多种类型模型的综合服务。

知识服务器中可以集中多种智能问题的知识库，或者是不同知识表示形式的知识(规则知识、谓词知识、框架知识、语义网络知识等)和多种不同的推理机，如正向推理机、逆向推理机、混合推理机等。

决策支持系统的综合部件(问题综合与交互系统)是由网络上的客户机来完成的，即在客户机上编制 DSS 控制程序，由它来调用或者组合模型服务器上的模型，完成模型计算；知识服务器上的知识完成知识推理以及数据仓库的综合信息查询，或用历史数据进行预测。这样，就形成了网络环境的综合决策支持系统，其结构图如图 12.5 所示。

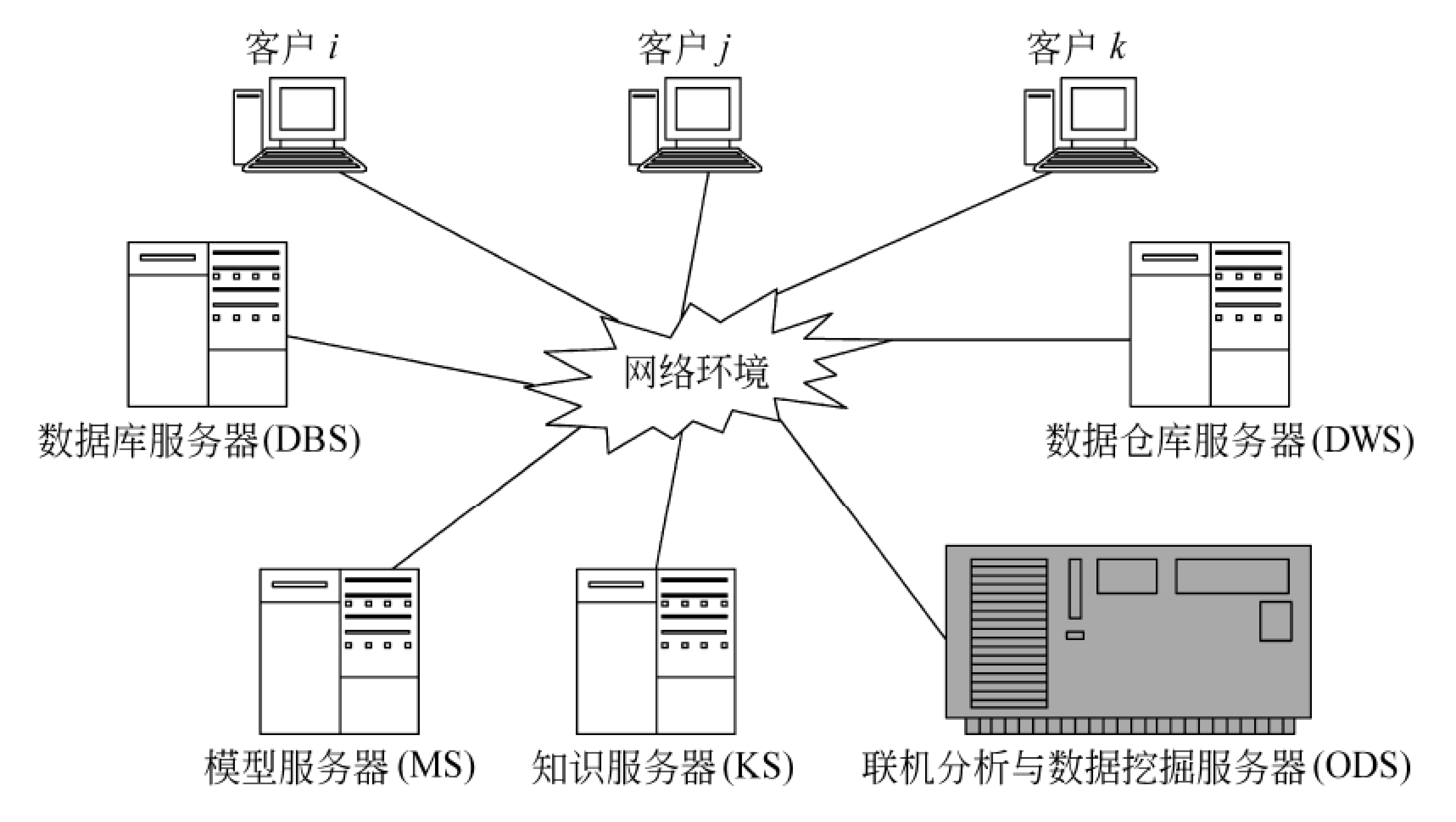


图 12.5　网络环境的综合决策支持系统结构

网络环境的综合决策支持系统是决策支持系统的发展方向。由于 Internet 技术的成熟和普及，这种结构形式的决策支持系统很快就会出现。

12.2 可拓数据挖掘

12.2.1 可拓学基本原理

具体来说，可拓学的理论和方法就是通过可拓变换与可拓知识来改变问题的目的或条件，去解决矛盾的问题。可拓学是我国学者蔡文教授提出的原创性理论和方法。

可拓学的详细内容参见《可拓逻辑初步》等书，在此只作简单说明。

1. 可拓学的基础信息

可拓学将客观世界的物、事、关系表示为物元、事元、关系元，把它们统称为基元，它们是可拓学的基础信息。

(1) 物元

物元表示为：$R=(N,c,v)$，是物 N、特征 c 及取值 v 的三元组。

例如：

$$R=(N,c,v)=\begin{bmatrix} N & c_1 & v_1 \\ & c_2 & v_2 \\ & c_3 & v_3 \end{bmatrix}=\begin{bmatrix} \text{工件} & \text{长度} & \text{30cm} \\ & \text{直径} & \text{6cm} \\ & \text{重量} & \text{2kg} \end{bmatrix} \tag{12.1}$$

(2) 事元

事元表示为：$I=(d,b,u)$，是动词 d、特征 b 及取值 u 的三元组。

例如：

$$I=(d,b,u)=\begin{bmatrix} d & b_1 & u_1 \\ & b_1 & u_2 \\ & b_3 & u_3 \\ & b_4 & u_4 \end{bmatrix}=\begin{bmatrix} \text{打} & \text{支配对象} & \text{球} \\ & \text{施动对象} & \text{小明} \\ & \text{时间} & \text{下午} \\ & \text{地点} & \text{球场} \end{bmatrix} \tag{12.2}$$

(3) 关系元

关系元表示为：$Q=(s,A,W)$，是关系 s、特征 A 及量值 W 的三元组。

例如：

$$Q=(s,A,W)=\begin{bmatrix} s & a_1 & w_1 \\ & a_2 & w_2 \\ & a_3 & w_3 \\ & a_4 & w_4 \end{bmatrix}=\begin{bmatrix} \text{借贷} & \text{前项} & \text{公司 } A \\ & \text{后项} & \text{银行 } B \\ & \text{程度} & \text{100 万元} \\ & \text{维系方式} & \text{合同} \end{bmatrix} \tag{12.3}$$

2. 可拓变换

解决矛盾问题的工具是可拓变换。通过可拓变换，使求知问题中不可知问题变为可知问题，使求行问题中不可行问题转化为可行问题，使假命题变为真命题，即通过可拓变换变矛盾问题为不矛盾问题。

可拓变换是把一个对象变为另一个对象，即可拓变换 T 将基元 u 变成基元 v，表示为

$$Tu = v \tag{12.4}$$

可拓变换 T 包括置换、增加、删减、扩大、缩小等。具体表示为：

(1) 置换变换：$T(A)=A'$

(2) 增加变换：$T(A)=A\oplus A_l$

(3) 删减变换：$T(A)=A-A_l$

(4) 扩缩变换：$T(A)=\alpha A$，当 $\alpha>1$ 时为扩大变换，当 $0<\alpha<1$ 时为缩小变换。

3. 可拓信息

可拓信息是解决矛盾问题的信息。可拓学的基元(物元、事元、关系元)是可拓信息的基础信息。可拓学的变换是变化信息，通过变换才能变矛盾问题为不矛盾问题。

$$\text{可拓信息} = \text{基元(基础信息)} + \text{可拓变换(变化信息)} \tag{12.5}$$

可拓信息中的基元信息属于静态的描述，而变化信息属于变化的信息，具有变化特征。解决矛盾必须通过可拓变换，即利用变化的信息才能解决矛盾问题。

4. 可拓学的基础知识

可拓学的基础知识为拓展式，包括发散式、相关式、可扩式、蕴涵式等。

- 发散式：$(N_1,c_1,v_1)\dashv(N_1,c_1,v_i)$，　$i=1,2,\cdots,n$
- 相关式：$(N_1,c_1,v_1)\backsim(N_2,c_2,v_2)$
- 蕴涵式：$(N_1,c_1,v_1)\rightarrow(N_2,c_2,v_2)$
- 可扩式：$(N_1,c_1,v_1)\wedge(N_2,c_2,v_2)$

可拓学的传导原理是可拓变换 T_u 引起相应的传导变换 T_v，将这种关系表示为变换蕴涵式，它是变化的知识：

$$(T_u u = u')\rightarrow(T_v v = v'),\quad \text{简写为}\quad T_u\rightarrow T_v \tag{12.6}$$

5. 关联函数

可拓学引入关联函数，将矛盾问题进行量化处理，称为量化知识。

关联函数公式：

$$k(x) = \frac{\rho(x,X_0)}{D(x,X_0,X)} \tag{12.7}$$

其中

$$\rho(x,X_0) = \left|x-\frac{a+b}{2}\right|-\frac{b-a}{2} \tag{12.8}$$

ρ 表示点 x 到区间 $X_0=\langle a,b\rangle$ 之距，$D(x,X_0,X)=\rho(x,X)-\rho(x,X_0)$。$x$ 在 X_0 中，$k(x)>0$ 是正域区间，即量变区间。x 在区间 $X=\langle c,d\rangle$ 中，$k(x)<0$ 是负域区间，即质变区间。

关联函数本身属于知识。当 x 从区间 X_0 变化到区间 X 后，即关联函数 $k(x)$ 由正数变为负数，表明从量变到质变，即矛盾问题得到解决。

6. 可拓知识

在智能科学中，对知识概念有很多定义，比较典型的定义有：

(1) Feigenbaum(费根鲍姆)定义：知识是信息经过加工整理、解释、挑选和改造而形成的。

(2) Bernstein(Hayes-Roth 引用)定义：知识是某一特定域的表达式、关系和过程构成的。

(3)《人工智能辞典》中的定义：知识是人们对客观世界的规律性的认识。

可见，知识是对信息进行加工，得到如表达式、关系等规律性的信息。知识是对信息进行浓缩，找出事物中存在的规律。

可拓知识是解决矛盾问题的知识。可拓学的拓展式(发散式、相关式、可扩式、蕴涵式等)是可拓知识的基础知识。可拓学的传导原理的变换蕴涵式是变化知识。可拓学的关联函数将矛盾问题进行量化处理，称为量化知识。它们共同构成了可拓知识，即：

$$可拓知识 = 拓展式(基础知识) + 变换蕴涵式(变化知识) + 关联函数 \tag{12.9}$$

可拓知识中的拓展式中的蕴涵式与人工智能的产生式规则是一致的。拓展式中的发散式、相关式、可扩式等可以看成是产生式的扩展。它们具有静态特征。

可拓知识中的变换蕴涵式是典型的变化知识，是解决矛盾问题的更有价值的知识。可见，可拓知识体现了变化知识的特点。

12.2.2 从数据挖掘到可拓数据挖掘

数据挖掘是从数据中挖掘出知识。由于数据具有静态性，代表已存在的事实，所挖掘的知识也具有静态性。

我们提出可拓数据挖掘，在于挖掘可拓知识，是数据挖掘的扩展。主要包含如下两类：

1. 挖掘关联函数的区间信息

解决矛盾问题的量化方法是建立关联函数，通过可拓变换使变量 x 从区间 X_0 变换到 X，区间参数 a、b、c、d 一般是运用实验或统计得到。

利用数据挖掘方法获取区间参数信息，是可拓数据挖掘的一类重要任务。

2. 挖掘变换蕴涵式的可拓数据挖掘

数据挖掘中能获取知识(条件→结论)，对条件进行可拓变换和对结论进行传导变换，获得的变化的知识，即可拓知识：

$$T_{条件} \rightarrow T_{结论} \tag{12.10}$$

把这种挖掘变化的知识称为可拓数据挖掘。

12.2.3 可拓数据挖掘理论

1. 可拓数据挖掘定理

定理 1　对于两类规则

$$A \rightarrow P \tag{12.11}$$

$$B \rightarrow N \tag{12.12}$$

一般情况 $A=\wedge a_i, B=\wedge b_j$

若存在条件的可拓变换 $T_{条件}$：

$$T_{条件}(B) = A \tag{12.13}$$

并存在结论的可拓变换 $T_{结论}$（为 $T_{条件}$ 的传导变换）：

$$T_{结论}(N) = P \tag{12.14}$$

则成立可拓变换知识（变化知识）

$$T(B) = A \rightarrow T(N) = P \tag{12.15}$$

即

$$\text{if} \quad T(B) = A \quad \text{then} \quad T(N) = P \tag{12.16}$$

证明：

(1) 定理的已知条件表示成命题逻辑公式，并化为子句型：

① $A \rightarrow P \leftrightarrow \neg A \vee P$

② $B \rightarrow N \leftrightarrow \neg B \vee N$

③ $T(B)=A \leftrightarrow \neg B \wedge A \leftrightarrow \neg B, A$

④ $T(N)=P \leftrightarrow \neg N \wedge P \leftrightarrow \neg N, P$

(2) 对定理的结论取非后化成子句型：

$\neg(T(B)=A \rightarrow T(N)=P) \leftrightarrow \neg[(\neg B \wedge A) \rightarrow (\neg N \wedge P)] \leftrightarrow \neg[\neg((\neg B) \wedge A) \vee (\neg N \wedge P)] \leftrightarrow \neg[(B \vee \neg A) \vee (\neg N \wedge P)] \leftrightarrow \neg(B \vee \neg A) \wedge \neg(\neg N \wedge P) \leftrightarrow \neg B \wedge A \wedge (N \vee \neg P) \leftrightarrow \neg B, A, N \vee \neg P$

(3) 对全部子句集进行归结：

① 全部子句集为：$\neg A \vee P, \neg B \vee N, \neg B, A, \neg N, P, N \vee \neg P$。

② 归结过程：子句 $\neg A \vee P$ 与子句 A 归结为 P，它与子句 $N \vee \neg P$ 归结为 N，再和子句 $\neg N$ 归结为空子句。产生矛盾，故证明定理正确。

定理 2 对于两条同类规则

$$A \rightarrow P \tag{12.17}$$

$$C \wedge B \rightarrow P \tag{12.18}$$

若存在可拓变换

$$T(B) = A \tag{12.19}$$

则成立可拓变换知识

$$T(B) = A \rightarrow P \tag{12.20}$$

即

$$\text{if} \quad T(B) = A \quad \text{then} \quad P \tag{12.21}$$

该定理同样可用归结原理证明，在此省略。

2. 可拓数据挖掘过程

从可拓数据挖掘定理中可以概括可拓数据挖掘过程为：

步骤 1：对分类问题利用数据挖掘方法获得分类规则，即获得公式(12.11)和(12.12)的知识。

步骤 2：确定规则的前提中存在可拓变换以及结论中存在可拓变换，即找出满足公式

(12.13)和(12.14)的可拓变换。

步骤 3：利用定理 1 和定理 2 获得可拓知识(12.15)或(12.20)。

3. 可拓推理

在智能科学中，知识推理采用了形式逻辑中的假言推理。即

$$P \wedge (P \rightarrow Q) \vdash Q \tag{12.22}$$

可拓推理是对拓展式和变换蕴涵式的假言推理。

(1) 拓展推理

对拓展式的假言推理称为拓展推理。以发散式为例，发散式推理表示为

$$(N_1, c_1, v_1) \wedge [(N_1, c_1, v_1) \dashv (N_1, c_1, v_i)] \vdash (N_1, c_1, v_i) \tag{12.23}$$

(2) 传导推理

变换蕴涵式是可拓变换与传导变换之间的蕴涵式，它的假言推理称为传导推理，表示为

$$(Tu = u') \wedge [(Tu = u') \rightarrow (Tv = v')] \vdash (Tv = v') \tag{12.24}$$

可拓推理是在知识推理的基础上扩展为对变化知识的推理。

下面证明可拓推理公式(12.24)是正确的。

证明：

(1) 将公式(12.24)中推理($\vdash$)的左部写成等价的命题逻辑公式：

$$(\neg u \wedge u') \wedge [(\neg u \wedge u') \rightarrow (\neg v \wedge v')]$$

(2) 上式化为子句型：

$(\neg u \wedge u') \wedge [(\neg u \wedge u') \rightarrow (\neg v \wedge v')] \leftrightarrow (\neg u \wedge u') \wedge [\neg(\neg u \wedge u') \vee (\neg v \wedge v')] \leftrightarrow (\neg u \wedge u') \wedge [(u \vee \neg u') \vee (\neg v \wedge v')] \leftrightarrow (\neg u \wedge u') \wedge [(u \vee \neg u' \vee \neg v) \wedge (u \vee \neg u' \vee v')] \leftrightarrow (\neg u \wedge u') \wedge (u \vee \neg u' \vee \neg v) \wedge (u \vee \neg u' \vee v') \leftrightarrow \neg u, u', (u \vee \neg u' \vee \neg v), (u \vee \neg u' \vee v')$

(3) 将推理($\vdash$)的右部取非后，化为子句型：

$$\neg(Tv = v') \leftrightarrow \neg(\neg v \wedge v') \leftrightarrow v \vee \neg v'$$

(4) 归结过程：子句 $v \vee \neg v'$ 与子句$(u \vee \neg u' \vee \neg v)$归结为$\neg v' \vee u \vee \neg u'$，它与子句$\neg u$ 归结为$\neg v' \vee \neg u'$，与 u'归结为$\neg v'$，再与子句$(u \vee \neg u' \vee v')$归结为 $u \vee \neg u'$，与$\neg u$ 归结为$\neg u'$，再与 u'归结为空子句。

产生矛盾，证明可拓推理公式(12.24)是正确的。

可拓知识只表明存在变化的可能性。可拓推理表明实际变化的发生。在公式(12.15)中，可拓知识($T_u \rightarrow T_v$)表明对 u 的变换 T_u会引起对 v 的变换 T_v。而可拓推理式(12.24)表明现已发生变换 T_u，按公式(12.24)必然出现变换 T_v。

12.2.4 可拓数据挖掘实例

1. **实例** 1

气候训练集如数据表 12.1 所示。

表 12.1　气候训练集

No.	属　性				类　别
	天气	气温	湿度	风	
1	晴	热	高	无风	N
2	晴	热	高	有风	N
3	多云	热	高	无风	P
4	雨	适中	高	无风	P
5	雨	冷	正常	无风	P
6	雨	冷	正常	有风	N
7	多云	冷	正常	有风	P
8	晴	适中	高	无风	N
9	晴	冷	正常	无风	P
10	雨	适中	正常	无风	P
11	晴	适中	正常	有风	P
12	多云	适中	高	有风	P
13	多云	热	正常	无风	P
14	雨	适中	高	有风	N

(1) 数据挖掘获取的规则知识有(参见本书 7.2.2 小节)：

- if 天气＝晴 and 湿度＝正常 then 类别＝P
- if 天气＝多云 then 类别＝P
- if 天气＝雨 and 风＝无风 then 类别＝P
- if 天气＝晴 and 湿度＝高 then 类别＝N
- if 天气＝雨 and 风＝有风 then 类别＝N

(2) 存在的可拓变换：

① 条件变换

- T_1(天气＝晴)＝(天气＝多云)
- T_2(天气＝晴)＝(天气＝雨)
- T_3(天气＝雨)＝(天气＝多云)
- T_4(天气＝多云)＝(天气＝晴)
- T_5(天气＝雨)＝(天气＝晴)
- T_6(天气＝多云)＝(天气＝雨)
- T_7(湿度＝高)＝(湿度＝正常)
- T_8(湿度＝正常)＝(湿度＝高)
- T_9(风＝无风)＝(风＝有风)
- T_{10}(风＝有风)＝(风＝无风)

② 结论变换

- T(N)＝P
- T(P)＝N

(3) 利用可拓数据挖掘定理 1 和定理 2 可以得到的变化知识：

① 类别发生变化的知识

- （天气＝晴） and （T_7(湿度＝高)＝(湿度＝正常))→T(N)＝P
- （湿度＝高） and （T_1(天气＝晴)＝(天气＝多云))→T(N)＝P
- （天气＝雨） and （T_{10}(风＝有风)＝(风＝无风))→T(N)＝P
- （风＝有风） and （T_3(天气＝雨)＝(天气＝多云))→T(N)＝P
- （天气＝晴） and （T_8(湿度＝正常)＝(湿度＝高))→T(P)＝N
- （天气＝雨） and （T_9(风＝无风)＝(风＝有风))→T(P)＝N

② 类别不发生变化的知识

- （湿度＝正常） and （T_1(天气＝晴)＝(天气＝多云))→类别＝P
- （风＝无风） and （T_3(天气＝雨)＝(天气＝多云))→类别＝P
- （风＝无风） and （T_6(天气＝多云)＝(天气＝雨))→类别＝P
- （湿度＝正常） and （T_4(天气＝多云)＝(天气＝晴))→类别＝P

2. 实例 2

在“脑血栓”与“脑出血”两类疾病的数据库中进行数据挖掘和可拓数据挖掘。

(1) 在数据库中通过数据挖掘获取规则知识

从“脑出血”和“脑血栓”两种疾病的大量实例数据库中，通过数据挖掘的遗传算法可以获取两种疾病独立诊断的规则知识(参见本书 10.5.3 小节)。获得的主要 7 条规则：

① 高血压＝有∧瞳孔不等大＝是∧膝腱反射＝不活跃 → 脑出血
② 瞳孔不等大＝是∧语言障碍＝是 → 脑出血
③ 高血压＝有∧起病方式＝快∧意识障碍＝深度 → 脑出血
④ 高血压＝有∧病情发展＝快 → 脑出血
⑤ 高血压＝有∧动脉硬化＝有∧起病方式＝慢 → 脑血栓
⑥ 动脉硬化＝有∧病情发展＝慢 → 脑血栓
⑦ 动脉硬化＝有∧意识障碍＝无 → 脑血栓

(2) 确定存在的可拓变换

在医疗中病人存在的可拓变换有：

T(起病方式慢)＝起病方式快；T(无意识障碍)＝深度意识障碍。也存在可拓变换：T(脑血栓)＝脑出血。

(3) 利用可拓数据挖掘定理获取可拓知识(变化的知识)

根据定理 1 得到可拓变换知识(变化知识)为

T(有动脉硬化∧起病方式慢∧无意识障碍)＝起病方式快∧有深度意识障碍

$$\rightarrow T(\text{脑血栓}) = \text{脑出血} \qquad (12.25)$$

还可以得出其他的可拓知识。

3. 可拓推理应用

可拓知识(变化的知识)只说明前提的可拓变换会引起结论的可拓变换(传导变换)，并不表示已经发生了变化。可拓知识中的前提一旦在现实中出现，就可以利用可拓推理判断

可拓知识中结论的出现。

在实例 2 中，当发现某病人由起病方式慢变成起病方式快，同时无意识障碍变成有深度意识障碍，即可拓知识(12.25)的前提已经出现，利用可拓推理(12.24)就可以判断可拓知识(12.25)的结论已经出现，即应该诊断该病人已经由“脑血栓”变成了“脑出血”。治疗方式就应改由“脑血栓”的治疗方法变成治疗“脑出血”的方法。

两种疾病的治疗方法是完全相反的，若仍然用“脑血栓”的治疗方法治疗“脑出血”，将会快速加重“脑出血”症状，甚至于导致死亡。这条变化知识对医生来讲是极其重要的。

可见，挖掘变化知识的可拓数据挖掘比挖掘静态知识的数据挖掘更有意义。

习　题

1. 管理科学是如何辅助决策的？
2. 决策支持系统的结构是什么？它是如何辅助决策的？
3. 智能决策支持系统的结构是什么？
4. 数据仓库是如何辅助决策的？
5. 基于数据仓库的决策支持系统结构是什么？它是如何辅助决策的？
6. 新决策支持系统与传统决策支持系统有什么不同？
7. 为什么要把新决策支持系统与传统决策支持系统结合起来？
8. 综合决策支持系统的结构是什么？它由哪些主体组成？
9. 举例说明数据仓库中增加模型库如何提高辅助决策能力的。
10. 网络上的服务器是如何提高服务效果的？
11. 网络环境的综合决策支持系统的结构是什么？
12. 可拓学的基础信息有哪些？
13. 可拓变换有哪些？
14. 可拓信息如何定义？它的特点是什么？
15. 可拓学的基础知识有哪些？
16. 可拓学的关联函数如何定义？
17. 可拓知识如何定义？它与一般的知识定义有什么不同？
18. 什么是可拓数据挖掘？它与数据挖掘有什么不同？
19. 可拓数据挖掘的定理 1 是什么？它说明了什么？
20. 可拓数据挖掘的定理 2 是什么？它说明了什么？
21. 形式逻辑的假言推理公式是什么？
22. 可拓推理的公式是什么？它说明了什么？
23. 从实例 1 中得到的可拓知识有什么用？
24. 从实例 2 中得到的可拓知识有什么用？
25. 从实例中说明可拓数据挖掘与数据挖掘的区别和意义。

参考文献

[1] Inmon W H. 数据仓库. 北京:机械工业出版社,2000.

[2] Inmon W H,Rudim K. 数据仓库管理. 北京:电子工业出版社,2000.

[3] Ponniah P. 数据仓库基础. 北京:电子工业出版社,2004.

[4] 王珊等. 数据仓库技术与联机分析处理. 北京:科学出版社,1998.

[5] Bischoff J,Alexander T. 数据仓库技术. 北京:电子工业出版社,1998.

[6] Thomsen E. OLAP 解决方案:创造多维信息系统(第二版). 北京:电子工业出版社,2004.

[7] Linoff G S,Micheal S,Berry J A. Web 数据挖掘,将客户数据转化为客户价值. 北京:电子工业出版社,2004.

[8] 陈文伟,黄金才. 数据仓库与数据挖掘. 北京:人民邮电出版社,2004.

[9] 陈文伟,黄金才,赵新昱. 数据挖掘技术. 北京:工业大学出版社,2002.

[10] 陈文伟. 决策支持系统及其开发(第二版). 北京:清华大学出版社,2000.

[11] 陈文伟. 决策支持系统教程. 北京:清华大学出版社,2004.

[12] 陈文伟. 智能决策技术. 北京:电子工业出版社,1998.

[13] Han J,Kamber M. 数据挖掘概念与技术. 北京:机械工业出版社,2001.

[14] Kantardzic M. 数据挖掘——概念、模型和算法. 北京:清华大学出版社,2003.

[15] 徐立本. 机器学习引论. 长春:吉林大学出版社,1991.

[16] Mchalski R S,Garbonell J G,Mitchell T M. 机器学习实现人工智能的途径. 北京:科学出版社,1992.

[17] 洪家荣. 归纳学习——算法理论应用. 北京:科学出版社,1997.

[18] 史忠植. 知识发现. 北京:清华大学出版社,2002.

[19] 徐洁磬. 数据仓库与决策支持系统. 北京:科学出版社,2005.

[20] Marakas G M. 21 世纪的决策支持系统. 北京:清华大学出版社,2002.

[21] 刘请. Rough 集及 Rough 推理. 北京:科学出版社,2001.

[22] 王国胤. Rough 集推理论与知识获取. 西安:西安交通大学出版社,2001.

[23] 蔡文,杨春燕,何斌. 可拓逻辑初步. 北京:科学出版社,2003.

[24] 陈文伟等. 数据仓库与决策支持系统. 计算机世界,1998-06-15.

[25] 高人伯,陈文伟. 数据仓库和 OLAP 的数据组织. 计算机世界,1998-06-15.

[26] 黄金才,陈文伟,陈元. 数据仓库中的元数据. 计算机世界,1998-06-15.

[27] 陈元,陈文伟. OLAP 的多维数据分析. 计算机世界,1998-06-15.

[28] 陈文伟等. 综合决策支持系统. 计算机世界,1998-06-15.

[29] 陈文伟等. 数据开采与知识发现综述. 计算机世界,1997-06-30.

[30] 陈文伟,钟鸣. 数据开采的决策树方法. 计算机世界,1997-06-30.

[31] 马建军,陈文伟. 数据开采的集合论方法. 计算机世界,1997-06-30.

[32] 邹雯,陈文伟. 数据开采中的遗传算法. 计算机世界,1997-06-30.

[33] 王珊,罗立. 从数据库到数据仓库. 计算机世界,1996-07-15.

[34] 王珊. 数据仓库、联机分析处理、数据挖掘——基于数据库技术的 DSS 解决方案. 计算机世界,1997-01-06.

[35] 陈文伟,黄金才等. 决策支持系统新结构体系. 管理科学学报,1998-09.

[36] 陈文伟,赵东升等.医疗事故(事件)辅助鉴定与管理系统.计算机工程与应用,1999-07.
[37] 徐立本,陈文伟. 机器学习论文集.国防科技大学学报,1995 17.增刊.
[38] 钟鸣,陈文伟.示例学习的抽象信道模型及其应用.计算机研究与发展,1992,29 (1).
[39] 钟鸣,陈文伟.示例学习算法 IBLE 和 ID3 的比较研究.计算机研究与发展,1993,30 (1).
[40] 陈文伟等.数据开采技术研究.清华大学学报(自然科学版),1998,38 (2).
[41] 马建军,陈文伟.关于集合理论的 KDD 方法.计算机应用研究,1997,14 (3).
[42] 陈文伟,黄金才.基于神经网络的模糊推理.模糊系统与数学,1996 (4).
[43] 邹雯,陈文伟.一种新的遗传分类器学习系统(GCLS)96'人工智能进展,1996.
[44] 陈文伟,钟鸣,赵东升.示例学习的信息理论以及逻辑公式的生成.中国机器学习 93'论文集,1993.
[45] 赵新昱,陈文伟,何义.基于算子空间的公式发现算法研究.国防科技大学学报,2000,22 (4):51-56.
[46] 赵新昱,陈文伟.基于遗传建模的公式发现研究.计算机工程与科学,2000,22 (5):84-87.
[47] 赛英,陈文伟等.从数据库中发现知识的方法研究与应用.管理科学学报,1999 年 9 月.
[48] 陈文伟,张帅等.经验公式发现系统 FDD.小型微型计算机系统,1999 年 6 月.
[49] 陈文伟.可拓学与智能科学、信息科学.香山科学会议(第 271 次会议),2005.12.
[50] 陈文伟,黄金才.从数据挖掘到可拓数据挖掘.中国人工智能进展,2005.
[51] 王颖楠等.Web 挖掘技术.吉林工学院学报,2002,23 (1).
[52] 韩家炜等.Web 挖掘研究.计算机研究与发展,2001,38 (4).
[53] 陈莉,焦李成.Web 数据挖掘研究现状及最新进展.西安电子科技大学学报(自然科学版),2001,28 (1).
[54] 胥桂仙等.文本挖掘中的特征表示及聚类方法.吉林工学院学报,2002,23 (3).
[55] 李凡等.关于文本特征抽取新方法的研究.清华大学学报(自然科学版),2001,41 (7).
[56] Barquin R C,Edelstein H A. Planning and Designing the Data Warehouse. Prentice Hall PTR,1997.
[57] Michalski R S,et al. Machine Learning an artificial intelligence approachn. Morgan Kaufmenn,1986.
[58] Quinlan J R. Induction of Decision Trees. Machine Learning,1986,1 (1).
[59] Quinlan J R. $C_{4.5}$:Program for Machine Learning. Margan Kovnfmenn Publishers,1993.
[60] Cendrowska J. PRISM: An Algorithm for inducing modular rules. Int. J. Man-Machine studies,1987,27.
[61] Adriaans P,Zantinge D. Data Mining,Addison-Wesley,1996.
[62] Fayyad U, Pietetsky-Shapiro G,Smyth P. From Data Mining to Knowledge Discovery in Database. AAAI,1996,0738-4602.
[63] Yasdi R. Learning Classification Rules from Database in the Context of Knowledge Acquisition and Representation. IEEE TKDE,1991,3 (3).
[64] Fayyad U,Uthurusamy R. Proc. of the First Int. Conf. on Knowledge Discovery and Data Mining. AAAI Press,1995.
[65] Zou Wen,Chen Wenwei. A New Genetic Classifier Learning System (GCLS). Genetic Programming Conference(GP'97),1997.
[66] Shi Zhongzhi. Principles of Machine Learning. International Academic Publishers,1992.
[67] Fayyad U, Piatetsky-Shapiro G. Advances in Knowledge Discovery and Data Mining. AAAI Press,1996.
[68] Fayyad U,Uthurusamy R. Data Mining and Knowledge Discovery in Database. Communications of the ACM,1996,39 (11).

高等院校信息管理与信息系统专业系列教材

书名	作者	定价
运筹学教程	刘满凤　等	43
信息系统开发方法教程(第三版)	陈佳	24
信息系统开发方法教程(第三版)题解与实验指导	陈佳	19
计算机组成原理教程(第 4 版)	张基温	25
计算机组成原理教程习题解析	张基温、孙仲美	16
离散数学(第四版)	耿素云、屈婉玲	24
离散数学题解(第三版)(与《离散数学(第四版)》配套)	耿素云、屈婉玲	18
数据结构及应用算法教程	严蔚敏	29
数据库系统原理教程	王珊、陈红	18.5
电子商务概论(第 3 版)	方美琪	49
社会统计分析及 SAS 应用教程	蔡建琼 等	26
信息系统开发与管理教程(第二版)	左美云	28
管理信息系统教程(第二版)	闪四清	29
电子商务基础教程(第二版)	兰宜生	32
信息资源管理教程	赖茂生	32
信息经济学教程	陈禹	17
数据仓库与数据挖掘教程	陈文伟	25
计算机网络教程(第二版)	黄叔武	29.8
计算机操作系统教程	张不同	27.5
信息系统分析与设计	杨选辉	29
信息系统安全教程	张基温	23
信息管理学教程(第 3 版)	杜栋	25
运筹学模型与方法教程	程理民	21
运筹学模型与方法教程例题分析与题解	刘满凤	22
决策支持系统教程	陈文伟	28
信息管理英语教程	李季方	26
Visual Basic 程序开发教程	张基温	26
Visual Basic 程序开发例题与题解	张基温	18
C++ 程序开发教程	张基温	26
C++ 程序开发例题与习题	张基温	26
Java 程序开发教程	张基温	24
Java 程序开发例题与习题	张基温	24

读者意见反馈

亲爱的读者：

感谢您一直以来对清华版计算机教材的支持和爱护。为了今后为您提供更优秀的教材，请您抽出宝贵的时间来填写下面的意见反馈表，以便我们更好地对本教材做进一步改进。同时如果您在使用本教材的过程中遇到了什么问题，或者有什么好的建议，也请您来信告诉我们。

地址：北京市海淀区双清路学研大厦 A 座 602　计算机与信息分社营销室 收
邮编：100084　电子邮箱：jsjjc@tup.tsinghua.edu.cn
电话：010-62770175-4608/4409　邮购电话：010-62786544

教材名称：算法设计与分析

ISBN：7-302-13154-6/TP • 8320

个人资料

姓名：________ 年龄：______ 所在院校/专业：____________

文化程度：________ 通信地址：____________

联系电话：________ 电子信箱：____________

您使用本书是作为：□指定教材 □选用教材 □辅导教材 □自学教材

您对本书封面设计的满意度：

□很满意 □满意 □一般 □不满意　改进建议____________

您对本书印刷质量的满意度：

□很满意 □满意 □一般 □不满意　改进建议____________

您对本书的总体满意度：

从语言质量角度看 □很满意 □满意 □一般 □不满意

从科技含量角度看 □很满意 □满意 □一般 □不满意

本书最令您满意的是：

□指导明确 □内容充实 □讲解详尽 □实例丰富

您认为本书在哪些地方应进行修改？（可附页）

____

____

您希望本书在哪些方面进行改进？（可附页）

____

____

电子教案支持

敬爱的教师：

为了配合本课程的教学需要，本教材配有配套的电子教案（素材），有需求的教师可以与我们联系，我们将向使用本教材进行教学的教师免费赠送电子教案（素材），希望有助于教学活动的开展。相关信息请拨打电话 010-62776969 或发送电子邮件至 jsjjc@tup.tsinghua.edu.cn 咨询，也可以到清华大学出版社主页（http://www.tup.com.cn 或 http://www.tup.tsinghua.edu.cn）上查询。